PEMBANGUNAN SISTEM APLIKASI HANASU : PEMBELAJARAN BAHASA JEPANG ANDROID MOBILE MEMANFAATKAN GOOGLE SPEECH RECOGNITION LIBRARY

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Eko Prasetyo Lukman Nur Hakim
NIM: 145150200111086

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

BAB 1 PENDAHULUAN

Bab pertama disini menjelaskan latar belakang yang menjadi pokok permasalahan yang akan diselesaikan pada tahap akhir, diharapkan penelitian ini dapat menyelesaikan permasalahan yang diangkat dengan mencari rumusan masalah, tujuan, batasan dan manfaat dari permasalahan dan solusi yang diangkat.

1.1 Latar belakang

Indonesia merupakan negara kedua di dunia untuk jumlah penduduk yang mempelajari Bahasa Jepang, tercatat sebanyak 872,406 orang yang mempelajari bahasa Jepang di Indonesia pada tahun 2012 dibawah negara China dengan jumlah orang 1,406,490 orang, tentunya 5 tahun berlalu jumlah peminat Bahasa Jepang terus meningkat. Pembelajar Bahasa Jepang di Indonesia didominasi oleh siswa menengah dengan persentase sebesar 40% (Tadashi, 2015). Berdasarkan alasan tersebut penelitian ini dibuat karena terdapat peluang untuk membantu banyak orang khususnya yang ingin belajar Bahasa jepang. Karena dengan belajar Bahasa Jepang terdapat banyak keuntungan seperti, untuk mendapatkan beasiswa, pekerjaan dan peluang bisnis dimana masyarakat Jepang terkenal konsumtif dengan rata orang dewasa aktif menghabiskan uang sebanyak 4.7 juta yen (Rp 508 juta) sehingga dapat menjadi peluang bagi eksportir luar negeri (Adi, 2012). Bahasa Jepang sendiri di Indonesia hanya dapat dipelajari di sekolah dan tempat belajar kursus, sehingga membuat beberapa masyarakat yang ingin dan minat mempelajari Bahasa Jepang belajar secara otodidak. Ada banyak cara untuk belajar Bahasa Jepang secara otodidak, bisa melalui buku, web, atau aplikasi yang ada di *Smartphone*. Jumlah penduduk Indonesia yang ingin belajar Bahasa Jepang dapat dilihat pada Tabel 1.1.

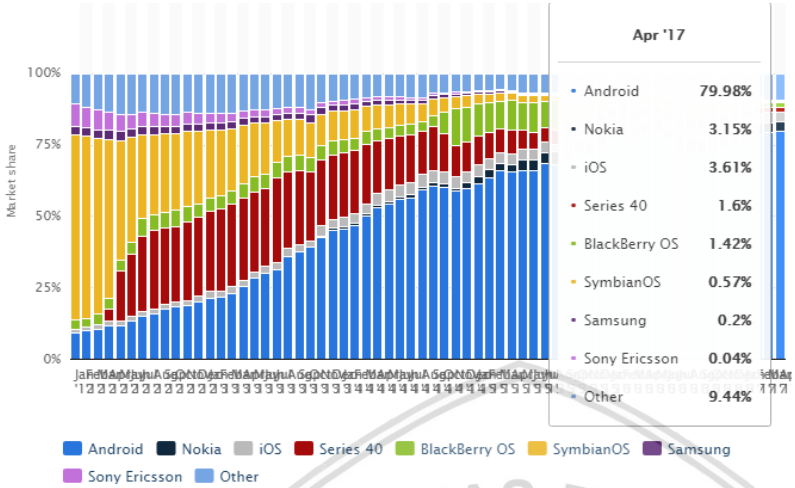
Tabel 1.1 Survey The Japan Foundation (2012)

	Tahun 2009	Tahun 2012
Lembaga	1,988	2,346
Pengajar	4,089	4,538
Pembelajar	716,353	872,406

Dengan jumlah 2.346 lembaga yang disebutkan pada Tabel 1.1, 71.6 % lembaga menyatakan “Fasilitas dan peralatan yang kurang” , namun permasalahan tersebut cenderung berkurang. Di lain pihak, banyak juga lembaga yang menyatakan “kekurangan informasi bahan ajar dan metode pengajaran” (61.4%) dan “kekurangan bahan ajar” (41.0%). *(Japan Foundation* Jakarta, 2013). *The Japan Foundation* sendiri merupakan badan hukum yang bertugas memperkenalkan budaya Jepang ke seluruh dunia internasional yang didirikan pada tahun 1972 dibawah naungan Departemen Luar Negeri Jepang (*The Japan Foundation*, 1972).

Dengan adanya internet yang mudah diakses dimana-mana kita dapat mendapatkan informasi yang kita perlukan kapanpun kita mau. Internet dapat kita akses melalui komputer, laptop, ataupun *smartphone*. Tentu saja aplikasi dan web merupakan pilihan favorit untuk belajar Bahasa Jepang secara otodidak. Tercatat 63,1 juta orang atau sekitar 47,6 persen

mengakses internet dari ponsel pintar (Izza, 2016). Salah satu perangkat yang banyak digunakan di Indonesia adalah Android, menurut data statistik di Indonesia hampir 80% OS yang digunakan adalah android. Diagram penggunaan OS pada *smartphone* di Indonesia ditunjukkan pada Gambar 1.1.



Gambar 1.1 Statistik OS *Smartphone* di Indonesia, 2017

Aplikasi yang akan dibuat oleh penulis adalah aplikasi menghafal aksara dan kata-kata bahasa Jepang dengan mengucapkan langsung untuk latihan dan menjawab soal-soal yang akan diberikan. Kelebihannya adalah pelajar memperoleh kecakapan motorik seperti melafalkan kata-kata atau kalimat (Abdul Kadir Arno, 2014). Penulis menggunakan metode belajar berbicara karena berdasarkan riset cara terbaik untuk belajar bahasa asing adalah berbicara dan mendengarkan, karena dengan seperti itulah kita belajar bahasa ibu selama ini (Teresa Pica, 2012). Dan kita tidak akan pernah melupakannya, mungkin beberapa kata yang jarang kita gunakan. Ketika kita berbicara sebuah bahasa, kita perlu belajar terlebih dahulu, mendengarkan, dan mereproduksi untuk menjadi sebuah kalimat.

Aplikasi belajar Bahasa Jepang berbasis mobile sudah ada, aplikasi yang ada saat ini adalah aplikasi belajar menghafal aksara Jepang baik dalam bentuk Hiragana, Katakana, dan Kanji (Aplikasi "Cara Cepat dan Mudah Belajar Bahasa Jepang", 2017). Beberapa latihan untuk menghafal berada di dalamnya, pengguna dihadapkan dengan aksara Jepang, kemudian kita dituntut untuk menebak dalam Bahasa Latin dengan cara menekan salah satu pilihan jawaban. Kelemahannya adalah kita hanya memilih dan belajar dengan cara menghafal, sehingga sensor motorik pengguna kurang bekerja yang mengakibatkan hafalan kita hanya akan menjadi hafalan jangka pendek. Beberapa aplikasi juga ada yang menggunakan gambar sebagai media pembelajaran, dan pengguna diminta untuk menebak dalam Bahasa Jepang dengan cara menekan pilihan dari beberapa pilihan yang ada. (Aplikasi "Obenkyo", 2017) Kelebihannya kita dapat menghafal beberapa kosa-kata dengan melihat gambar, karena otak kanan lebih mudah membantu kita menghafal dibandingkan dengan otak kiri. Tidak sedikit pula ada aplikasi yang mengajarkan kita bagaimana menulis aksara Jepang menggunakan jari kita langsung di layar smartphone (Aplikasi "Write it! Japanese", 2017). Dengan ini kita bisa belajar menulis dengan baik dan benar, sensor motorik kita bekerja, sehingga kita dapat menghafal lebih baik. Namun saat ini hanya terbatas beberapa kata saja, bukan dalam bentuk kalimat. Aplikasi-aplikasi yang diambil adalah aplkasi yang berada dalam *top 5 google play store* yang berarti merupakan aplikasi yang memiliki jumlah download terbanyak dan jumlah *rate* tertinggi dengan kategori tertentu (Vincent, 2017).

Penulis memanfaatkan peluang untuk mengembangkan aplikasi berbasis mobile dengan OS Android supaya dapat membantu masyarakat Indonesia khususnya yang minat belajar berbahasa Jepang. Dengan mengembangkan metode belajar otodidak di *smartphone* dengan mengucapkan langsung kosakata Bahasa Jepang dengan memanfaatkan *Speech to Text Library* yang ada pada *smartphone* Android. Aplikasi yang akan penulis buat adalah aplikasi tentang belajar Bahasa Jepang sehari-hari mulai dari belajar huruf Jepang hingga kalimat yang digunakan sehari-hari dengan cara praktek berbicara langsung menggunakan Bahasa Jepang disertai gambar-gambar pendukung. Dengan ini pengguna dapat berinteraksi langsung sehingga pengguna akan terbiasa dan lebih mudah menghafal. Aplikasi yang hampir mirip adalah aplikasi Bahasa Jepang untuk kategori mahir dan tidak ada khususnya untuk Bahasa Indonesia.

Oleh sebab itu penulis membuat aplikasi "Hanasu" dan dibuat penelitian berjudul "PEMBANGUNAN SISTEM APLIKASI HANASU : PEMBELAJARAN BAHASA JEPANG BERBASIS ANDROID MOBILE MEMANFAATKAN GOOGLE SPEECH RECOGNITION LIBRARY" yang diharapkan menjadi solusi bagi masyarakat Indonesia yang ingin belajar Bahasa Jepang dengan efisien menggunakan smartphone mereka.

1.2 Rumusan masalah

Berdasarkan latar belakang yang dikaji diatas, penulis dapat merumuskan masalah yang disusun dibawah ini:

1. Bagaimana analisis kebutuhan, rancangan, implementasi dan hasil uji Aplikasi Hanasu menggunakan *speech-to-text library*?
2. Apakah aplikasi Hanasu dapat membuat pengguna belajar Bahasa Jepang dengan cepat dimanapun dan kapanpun dengan menggunakan suara ?

1.3 Tujuan

Tujuan dari penelitian ini adalah sebagai berikut :

1. Mengetahui cara untuk menganalisis, merancang, mengimplementasi dan menguji perangkat lunak menggunakan *speech-to-text library*.
2. Menyediakan sistem untuk dapat belajar Bahasa Jepang yang dapat digunakan dimanapun dan kapanpun.

1.4 Manfaat

1. Manfaat bagi penulis
   Dapat membangun sistem informasi sesuai dengan materi yang sudah dipelajari selama perkuliahan.
2. Manfaat bagi peneliti selanjutnya
   Dapat menjadi acuan untuk mengembangkan sistem informasi yang hampir sama.
3. Manfaat bagi pengguna
   Dapat belajar Bahasa Jepang dengan lebih efisien.

1.5 Batasan masalah

Penelitian ini memiliki beberapa batasan dalam pengembangannya, diantaranya :

1. Sistem yang akan dibuat hanya menggunakan Bahasa Indonesia dan Bahasa Jepang.
2. Sistem yang akan dibuat menggunakan konsep *Object Oriented.*
3. Dalam membangun sistem digunakan bahasa pemrograman Java serta XML dan menggunakan Android Studio 3.0.1.
4. Sistem yang akan dibuat hanya dapat dijalankan di *platform* Android dengan OS minimal Jellybean versi 4.0.
5. Sistem yang akan dibuat harus terkoneksi dengan internet.
6. *Speech to Text Library* yang akan digunakan adalah library milik *Google*.
7. Implementasi *speech-to-text* hanya akan menilai kebenaran cara melafalkan Bahasa Jepang.

1.6 Sistematika pembahasan

Penyusunan dokumen skripsi ini dibagi menjadi beberapa bab, yang terdiri dari :

1. BAB 1 – PENDAHULUAN

   Berisis tentang latar belakang masalah, rumusah masalah, tujuan penelitian, manfaat dari penelitian, batasan penelitian, dan sistematika pembahasa skripsi ini.

2. BAB 2 – LANDASAN KEPUSTAKAAN

   Berisi tentang kajian-kajian kepustakaan yang terhubung sehingga dijadikan referensi dalam pembuatan skripsi ini.

3. BAB 3 – METODOLOGI PENELITIAN

   Berisi tentang alur kerja penelitian yang dilakukan dan proses menyelesaikan masalah penelitian.

4. BAB 4 – ANALISIS KEBUTUHAN

   Berisis tentang segala hal yang terkait seputar proses penggalian keutuhan dalam proses penelitian serta berisikan tentang hal-hal yang akan berkaitan dengan perancangan dan pemodelan sistem berdasarkan data yang telah didapat di tahap analisis kebutuhan.

5. BAB 5 – PERANCANGAN DAN IMPLEMENTASI

   Berisi tentang hal-hal yang berkaitan dengan implementasi pengembangan sistem berdasarkan pemodelan yang sudah dilakukan di tahap pemodelan.

6. BAB 6 – PENGUJIAN

   Berisi tentang hal-hal yang berkaitan dengan pengujian sistem yang dilakukan oleh peneliti untuk menganalisis hasil yang telah didapat.

7. BAB 7 – PENUTUP

Berisi tentang hal-hal yang berisi kesimpulan dari penelitian yang telah dilakukan dan saran dalam pengembangan penelitian selanjutnya.

BAB 2 LANDASAN KEPUSTAKAAN

Pada bab ini berisikan pembahasan-pembahasan tentang teori dasar yang berelasi dengan penelitian yang dilakukan. Teori dasar yang akan dibahas pada bab ini yaitu penjelasan tentang konsep dasar rekayasa perangkat lunak, teknologi yang digunakan dalam penelitian, konsep dasar *Software Development Life Cycle* yang digunakan, dan teknik pengujian yang digunakan.

2.1 Rekayasa Perangkat Lunak

Rekayasa perangkat lunak adalah satu bidang profesi yang mendalami cara-cara pengembangan perangkat lunak termasuk pembuatan, pemeliharaan, manajemen organisasi pengembangan perangkat lunak dan manajemen kualitas. Rekayasa perangkat lunak adalah pengubahan perangkat lunak itu sendiri guna mengembangkan, memelihara, dan membangun kembali dengan menggunakan prinsip rekayasa untuk menghasilkan perangkatlunak yang dapat bekerja lebih efisien dan efektif untuk pengguna. Kriteria yang dapat digunakan sebagai acuan dalam merekyasa perangkat lunak :

1. Maintainability
2. Dependability
3. Robustness
4. Efficiency
5. Usability

2.2 Platform Android

Platform adalah arsitektur hardware/fondasi/standard bagaimana sebuah sistem dimana aplikasi/program dapat berjalan, atau bisa juga dikatakan platform adalah dasar dari teknologi dimana teknologi yang lain atau proses-proses dibuat. Sebuah platform terdiri dari sistem operasi yaitu program sistem koordinasi komputer yang memberikan perintah-perintah kepada prosesor dan hardware untuk melakukan operasi-operasi logis dan mengatur pergerakan data di komputer. Banyak orang beranggapan bahwa Platform dan Sistem Operasi adalah sama, namun pada kenyataanya tidak. Platform merupakan dasar atau tempat dimana sistem operasi bekerja, tanpa platform sistem operasi tidak akan bisa berjalan.

Android adalah sistem operasi Linux yang dirancang untuk perangkat bergerak layar sentuh seperti telepon pintar dan komputer tablet. Android awalnya dikembangkan oleh android.inc namun tahun 2005 Google membelinya yang ahirnya resmi dirilis pada tahun 2007. Sistem operasi ini memiliki banyak versi dengan menggunakan nama makanan ringan, seperti : Alpha, Beta, Cupcake, Donnut, Eclair, Froyo, Gingerbread, Honeycomb, Ice cream sandwich, Jellybean, Kitkat, Lollipop, Marsmellow, Nougat, dan terbaru Android O.

2.3 Bahasa Jepang

Bahasa Jepang adalah bahasa nomor 2 di Indonesia yang memiliki banyak peminat. Bahasa Jepang sendiri memiliki huruf yang unik karena tidak menggunakan Alphabet. Bahasa Jepang memiliki 3 macam jenis karakter, Hiragana, Katakana dan Kanji. Sebelum belajar Bahasa Jepang, diharuskan belajar mengetahui suku kata Bahasa Jepang untuk dapat belajar ke jenjang selanjutnya.

Diawali dengan membaca, menghafal, mengucap, dan barulah dapat mengerti Bahasa Jepang dengan sempurna.

2.4 Google Text-To-Speech Library

Google text to speech adalah library yang dibuat oleh Android untuk OS Android itu sendiri. Text to speech memiliki fungsi untuk membaca text yang berada di dalam layar kita. Ada lebih dari 110 macam bahasa yang ditawarkan oleh Google text-to-speech. Dengan kita mengimport library dari google ini kita tidak perlu lagi membuat pengaturan text-to-speech dari awal, kita hanya

2.5 Firebase

Firebase merupakan layanan database yang disediakan oleh Google dengan bermacam-macam fitur yang sudah siap tersedia. Beberapa fitur yang disajikan oleh Firebase *Realtime Database, Crashlytics, Authentfication, Cloud Storage, Android Test Lab ,* dan masih banyak lagi.

2.5.1 Realtime Database

Firebase menyediakan fitur untuk dapat menyimpan dan membaca data yang sudah disimpan dalam waktu milidetik. Dengan ini aplikasi dapat menyinkronkan data aplikasi secara waktu yang hampir bersamaan, tanpa perlu memuat ulang halaman.

2.5.2 Crashlytics

Crashlytics adalah fitur yang memungkinkan kita untuk memperbaiki masalah dengan *crash reporting* yang *realtime.* Dengan *crashlytics* kita dapat mengetahui berapa crash yang terjadi dalam satuan waktu yang kita inginkan.

2.5.3 Authentification

Firebase memiliki fitur untuk dapat menggunakan fungsi authentifikasi yang seluruh fungsi dari authentifikasi (masuk ke dalam sistem, keluar dari sistem, mendaftarkan diri ke dalam sistem, dan pengecekan apakah suda masuk ke dalam sistem atau belum) sudah disediakan oleh *Firebase*.

2.5.4 Cloud Storage

Firebase memiliki penyimpanan yang dapat digunakan untuk menyimpan berkas secara online. Sehingga kita tidak perlu untuk mengunggah berkas ke layanan penyimpanan lain.

2.5.5 Android Test Lab

Android test lab merupakan fitur untuk kita dapat menguji aplikasi kita berjalan di perangkat yang berbeda-beda. Firebase memiliki banyak sekali pilihan perangkat dengan berbeda-beda SDK dan ukuran layar. Dengan ini kita dapat mengetahui di perangkat mana saja aplikasi kita dapat atau tidak dapat berjalan dengan baik.

2.6 Agile Scrum

Scrum adalah salah satu metode rekayasa perangkat lunak dengan menggunakan prinsip-prinsip pendekatan *Agile*, yang bertumpu pada kekuatan kolaborasi tim, *incremental product* dan proses iterasi untuk mewujudkan hasil akhir.

Scrum sendiri bukan satu-satunya metode yang menggunakan pendekatan *Agile*. Terdapat juga metode *Extreme Programming* (XP) yang juga menggunakan pendekatan *Agile* dalam rekayasa perangkat lunak. Masing-masing metode memiliki fokus atau penekanan yang berbeda yang tentu saja dapat dikombinasika untuk menghasilkan proses yang optimal.

Teknik *Scrum* dapat dilakukan di sebuah kepanitiaan ataupun proyek lain di luar bisnis teknologi informasi. Dalam teknik *Scrum* terbagi dalam tiga peran, yang pertama adalah *Product Owner, Scrum Master* dan *Development/Scrum Team*. *Product owner* bertugas mengatur urusan dengan *Stakeholder* sedangkan *Scrum Master* mengurusi bagian internal, di bagian *Development Team* mengatur urusan teknik pengerjaan project dan pembahasan yag lebih rinci.

Banyak perusahaan multinasional menginternalisasi atau mengadopsi teknik ini sebagai standar bekerja mereka, karena secara umum teknik *Scrum* berhasil membuat beberapa perusahaan menaikan omset karena sistem yang telah teruji ini. Teknik *Scrum* membuat pekerjaan anda menjadi lebih tertata dan lebih detail.

Namun segala sesuatu memiliki nilai plun dan minus, begitu juga dengan teknik *Scrum* ini, *Scrum* menjadikan pekerjaan lebih rapi namun teknik ini tidak cocok diterapkan pada perusahaan jasa yang butuh deadline cepat. *Scrum* membuat pekerjaan menjadi lebih lama dalam estimasi waktu. Namun segala sesuatunya pasti dapat di selesaikan apabila sudah memiliki persiapan yang matang dari awalnya.

Teknik *Scrum* dimulai dari pembahasan project antara *Product Owner* dengan *Stakeholder* terkait, lalu dibentuklah *Scrum Master* dan *Development Team*. Langkah pertama yang dilakukan oleh *Scrum Master* adalah dengan membuat semua list pekerjaan yang sebut juga dengan *User Story*, *User Story* tersebut dibuat di media yang dinamakan *Backlog* yang di tempel di *TaskBoard*.

Setiap Backlog yang dibuat diberi estimasi dan standar atau tingkat kesulitan, anda bisa memakai angka 1, 2, 3 dan seterusnya untuk menjelaskan tingkat kesulitan maupun huruf seperti A, B, C dan lain sebagainya, dan yang paling penting dari kode ini adalah semua *development team* harus mengerti maksud kode tersebut. Setelah dibuat kode maka tugas selanjutnya membuat *sprint*, *sprint* adalah batasan waktu pengerjaan dan jika telah ditetapkan maka teknik *Scrum* tidak diberkenankan mengubah konsep ditengah jalan. (akhsanov-HRD Advess, 2016).

2.7 INVEST Mneomonic

User Story dibuat menggunakan *INVEST mneomonic.* Dimana *INVEST* diciptakan untuk mengingatkan bagaimana cara membuat *User Story* yang baik (Bill Awake, 2011). Dimana *INVEST* sendiri merupakan akronim dari :

1. I – Independent : setiap *story item* harus berdiri sendiri dan tidak lebih dari 1 statement.
2. N – Negotiable : *story item* yang bagus merupakan *story* yang dapat dinegosiasi, *story* yang mementingkan esensi daripada detail.
3. V – Valuable : *story item* harus bernilai dan berguna untuk pengguna. Meski terkadang *developer* memiliki *concern* terhadap suatu hal, tapi mementingkan keperluan pengguna tetap harus diutamakan.
4. E – Estimable : *story item* yang baik adalah *story* yang dapat diestimasi waktu pengerjaannya sehingga dapat membantu pengguna untuk men-*ranking* kebutuhannya.
5. S – Small : *story item* yang bagus adalah *story* yang kecil dalam estimasi sudut pandang penggunaan dalam hitungan minggu bahkan hari. Bukan *story* yang memandang "ini akan memakan waktu berbulan-bulan".
6. T – Testable : *story item* yang bagus tentunya merupakan *story* yang diperkirakan dapat dieksekusi dan dapat diuji meski kita belum memulai mengerjakannya.

Berdasarkan hasil wawancara yang dilakukan didapatkanlah sebuah *User Story*. Dalam penulisannya penelitian ini menggunakan format sederhana : *Sebagai <tipe pengguna >, saya ingin <tujuan> sehingga saya <alasan>*

2.8 A/B Testing

A/B Testing adalah metode pengujian usabilitas yang membandingkan 2 atau lebih variasi sebuah design / sistem untuk mencari mana yang performanya lebih baik. Dengan adanya membandingkan 2 halaman yang berbeda versi kita bisa dengan spesifik mengetahui dampak dari perubahan yang kita lakukan. Dibandingkan dengan metode lain , *A/B Testing* memiliki 4 keuntungan besar :

1. Merupakan cabang dari *website analytic*, dimana mengukur secara langsung perilaku dari pengguna di dunia nyata. Kita dapat dengan percaya diri memberi kesimpulan bahwa versi B lebih menjual daripada versi A , sehingga versi B lah yang akan diberikan ke seluruh pengguna di seluruh dunia.
2. Metode ini dapat mengukur performa yang terkecil sekalipun dengan statistic yang signifikan karena kita dapat melempar *traffic* pada setiap *design.*
3. Dapat menyelesaikan konflik tentang *guideline* yang bertentangan berdasarkan kualitas usabilitas dari sebuah sistem. Kita dapat melakukan *A/B Testing* dibawah Batasan kita sendiri.
4. Murah, metode ini dikatakan urah karena kita dapat membandingkan sebuah sistem dengan sistem yang sudah dibuat sebelumnya, kita dapat menaruh design baru ke server kecil lain kemudian kita tidak perlu memonitor secara terus-menerus.

2.9 Post-test Only Design Study

Post-test Only Design Study merupakan metode untuk membandingkan dua metode cara belajar untuk mencari manakah diantaranya yang lebih efisien. Saat ini penggunaan komputer sangat meningkat pesat. Edukasi elektronik kini berubah dari belajar langsung dari perangkat menuju lingkungan online didasarkan dengan kecepatan internet, aksesibilitas dan konektifitas.

Metode yang digunakan dibagi menjadi beberapa tahap, diantaranya *study design, participants, data collection, educational methods* dan hasil. *Study Design* adalah mempelajari lingkungan belajar yang saat ini akan diuji. *Participant* adalah memilih peserta yang akan mengikuti *post-test* berdasarkan kriteria yang sudah dipilih di awal agar memiliki tingkatan yang sama. *Data Collection* adalah tahap dimana menentukan aturan-aturan yang tidak boleh dilanggar agar pencapaian dari pengujian bsia tepat sasaran. *Educational Method* merupakan tahap untuk menjelaskan 2 method yang berbeda yang akan peserta alami. Setelah mendapatkan hasil maka ditarik kesimpulan.

2.10 Task Scenario

Cara paling efektif untuk mengetahui bagaimana sesuatu itu bekerja dan sesuatu itu tidak bekerja adalah dengan cara melihat orang lain menggunakannya. Ini adalah esensi dari pengujian *usability*. Untuk meneliti peserta kita biasanya memberikan mereka sesuatu untuk dilakukan. Beberapa tugas ini biasa disebut *task.* Dibandingkan dengan perintah sederhana seperti pengguna harus "X" tanpa penjelasan apapun, lebih baik mensituasikan beberapa permintaan dengan menggunakan skenario pendek yang cocok dengan *stage* untuk melakukan aksi dan menyediakan sedikit penjelasan da konteks mengapa pengguna harus melakukan "X".

Sebelum dapat menulis *task scenario* yang digunakan dalam testing , diharuskan untuk dapat menuliskan daftar tujuan umum yang harus dicapai. Contoh, "apa hal yang harus dilakukan setiap pengguna untuk menyelesaikan di web ini ?" , maka didapatkan 3 tujuan utama dalam web ini :

- Menemukan artikel dengan topik spesifik
- Mendafar untuk seminar mingguan tentang *user experience*
- Belajar tentang pelayanan konsultasi

Setelah mengetahui apa tujuan dari pengguna, maka barulah dapat memformulakan *task scenarios* menggunakan pendekatan pengujian *usability. Task Scenario* merupakan aksi dimana menanyakan peserta untuk dapat menguji beberapa antarmuka. Contoh, *task scenario* seperti :

> "Kamu merencanakan liburan ke NYC, 3 Maret – 14 Maret. Kau butuh membeli tiker pesawat dan hotel. Pergi ke website American Airlines dan jetBlue Airlines dan lihatlah mana yang memiliki harga paling murah."

Task Scenarios membutuhkan memberikan konteks sehingga pengguna dapat merasa ikut serta dalam antarmuka dan berpura-pura dalam bisnis atau personal *task* apabila mereka ada di rumah atau di kantor. Sayangnya penulisan *task* seringkali fokus terlalu

banyak dalam menyuruh pengguna untuk berinteraksi dengan fitur yang spesifik, dibandingkan dengan melihat jika atau bagaimana pengguna memilih untuk menggunakan antarmuka tersebut. Berikut merupakan 3 tips untuk menuliskan *task* unuk meningkatkan penglihatan *usability :*

1. **Make the Task Realistic**

   Menyuruh beberapa peserta untuk melakukan sesuatu yang normalnya mereka tidak melakukannya akan membuat mereka mencoba untuk menyelesaikan *task* tanpa benar – benar tau dengan antarmuka. Sayangnya penulisan *task* membuat itu lebih susah untuk peserta, contohnya peserta harus bebas untuk membandingkan produk – produk berdasarkan kriteria mereka sendiri.

   Bersaman dengan *task* yang realistis akan bergantung pada peserta yang direkrut dan pada fitur yang diuji. Contohnya, jika kita ingin menguji website hotel, maka kita perlu memastikan bahwa peserta merupakan anggota dari salah satu keluarga yang bertanggung jawab atas reservasi travel.

2. **Make the Task Actionable**

   Cara paling bagus adalah memerintahkan pengguna melakukan aksinya, daripada menanyakan mereka bagaimana cara mereka akan melakukannya. Jika kita menanyakan "Bagaimana cara kau melakukan X ?" atau "katakan padaku bagaimana kalian melakukan Y?" peserta biasanya akan menjawab menggunakan kata-kata bukan aksi. Sayangnya, data yang dilaporkan oleh mereka sendiri tidak begitu akurat saat mereka benar – benar menggunaka sistem.

   Kita dapat mengatakan bahwa task tidak begitu *actionable* ketika peserta mengatakan kepada fasilitator, menggunakan mouse kemudian berkata beberapa hal seperti ini "Aku akan klik ini pertama, kemudian akan menuju ke tempat dimana saya ingin pergi, dan aku akan meng-klik-nya".

3. **Avoid Giving Clues and Describing the Steps**

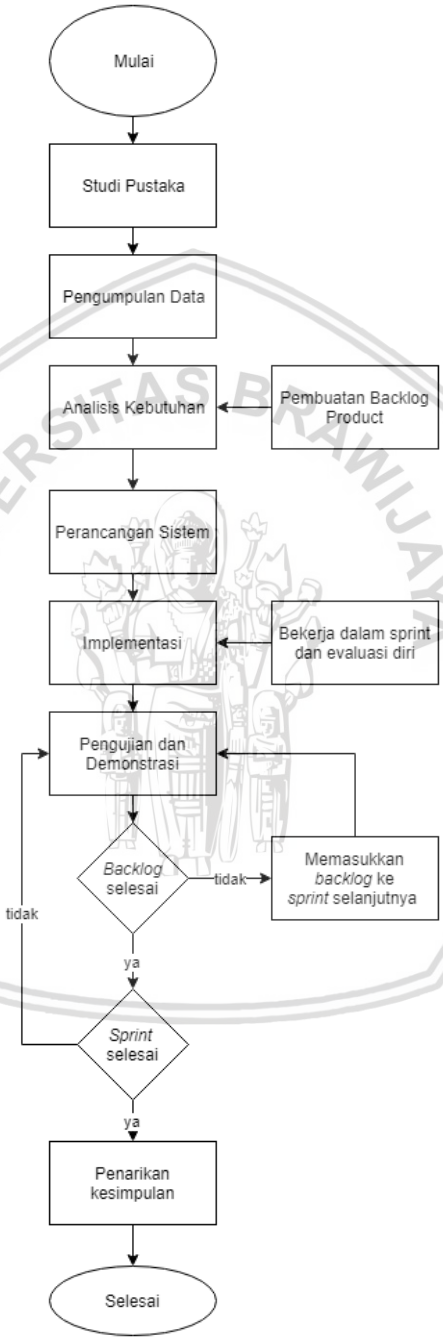
   Tahap – tahap deskripsi seringkali berisikan petunjuk tersembunyi sebagai bagaimana cara untuk menggunakan antarmuka. Contoh, jika kalian ingin mengatakan ke seseorang untuk meng-klik untuk keuntungan dalam *main menu,* kita tidak akan belajar bahwa menu label adalah sangat penting bagi mereka.

   Dengan menghindari memberi petunjuk bukan berarti membuat task yang ambigu. Contoh yang salah adalah "Buat perjanjian dengan dokter gigimu", dibandingkan dengan yang seperti itu lebih baik menggunakan kalimat "Buat perjanjian dengan doktermu Dr. Petersen pada hari Selasa pukul 10 pagi".

Jika *task scenario* terlalu ambigu, peserta biasanya akan menanyakan informasi lebih atau akan mengkonfirmasi apakah dia sudah berada di jalan yang tepat atau tidak, dimana seharusnya itu tidak perlu dilakukan.

BAB 3 METODOLOGI

Dalam pengembangan sebuah sistem diperlukan metode sebagai pedoman dalam merancang dan mengimplementasikannya pada sebuah sistem. Penelitian ini mengimplementasikan sebagian metode *Agile Scrum*, dengan mengganti proses meeting menjadi evaluasi diri. Metode pengembangan sistem digambarkan pada Gambar 3.1.



Gambar 0.1 Metode Pengembangan Sistem

3.1 Tipe Penelitian

Penelitian yang dilakukan ini merupakan penelitian implementatif, dimana peneliti akan membangun sistem belajar Bahasa Jepang di aplikasi android dan mewawancarai langsung tipe-tipe pengguna aplikasi.

3.2 Strategi dan Rancangan Penelitian

3.2.1 Subjek Penelitian

Subjek penelitian ini adalah murid SMA Negeri 1 Batu, dan guru Bahasa Jepang SMA Negeri 1 Batu.

3.2.2 Studi Pustaka

Studi pustaka menjelaskan dasar teori yang digunakan untuk menunjang penulisan skripsi. Teori-teori pendukung tersebut meliputi :

1. Rekayasa Perangkat Lunak
2. *Platform Android*
3. Bahasa Jepang
4. *Google Speech-to-Text Library*
5. *Firebase*
6. *Agile Scrum*
7. *INVEST Mneomonic*
8. *A/B Testing*
9. *Post-test Only Design Study*
10. *Task Scenario*

3.2.3 Teknik Pengumpulan Data

Metode pengumpulan data berdasarkan tujuannya dapat dibagi menjadi 2, yakni Kualitatif dan Kuantitatif. Kualitatif digunakan saat ingin mendapakan jawaban yang fleksibel dan bukan membutuhkan jawaban sederhana, sedangkan kuantitatif digunakan saat ingin survey sesuatu dan membutuhkan responden banyak dengan jawaban yang dapat diukur dengan angka. (Raluca Budiu, 2017)

Salah satu teknik pengumpulan data kualitatif adalah wawancara. Dalam penelitian ini menggunakan metode wawancara terhadap calon pengguna langsung. Penelitian ini memilih wawancara karena kita dapat menuju langsung calon pengguna kita dengan tepat dan dapat menggali lebih dalam apa yang mereka inginkan tanpa harus membuat suatu batasan. Wawancara yang baik adalah wawancara yang menggunakan narasumber antara 5 – 8 pengguna, karena kita dapat mengidentifikasi langsung permasalahan utama dalam sistem (Raluca Budiu, 2017). Di dalam penelitian ini 5 narasumber yang dipilih adalah

narasumber yang sedang menjalani kegiatan atau berkaitan dengan topik penelitian yang diangkat, diantaranya :

1. Sesi pertama wawancara dimulai dengan murid SMA Negeri 1 Batu yang sedang belajar mata pelajaran Bahasa Jepang dengan identitas dapat dilihat pada Tabel 3.1.

Tabel 0.1 Narasumber Murid SMA Negeri 1 Batu

No	Nama	Alamat	Umur
1	Maulidiya Selene S. P.	Perumahan Puri Indah Blok G2/7, Batu	17 Tahun
2	Maulina Isabel M. J.	Jl. Patimura gg 6 no 77, Kec Batu, Batu	17 Tahun
3	Amila Habsia	Jl Raya Diponegoro RT 4 RW 4 Banjar Tengah, Kec Dau, Malang	17 Tahun
4	Sanyyah Dewanti Putri	Jl Minsuarso no 44 RT 2 RW 13 Sisir, Kec Batu, Batu	16 Tahun

Sesi kedua dilanjutan dengan wawancara terhadap guru Bahasa Jepang SMA Negeri 1 Batu, dengan identitias pada Tabel 3.2

Tabel 0.2 Narasumber Guru Bahasa Jepang SMA Negeri 1 Batu

No	Nama	Alamat	Umur
1	Rini Wahyuningsih	Jl. Diran, Temas, Batu	41 Tahun

Dengan seluruh hasil wawancara yang akan dilampirkan di halaman lampiran.

2. Riset kedua yaitu riset kepustakaan, mencari sumber-sumber dari buku, artikel, e-book atau jurnal yang memiliki topik yang hampir sama dengan permasalahaan yang ingin diteliti.

   Dengan hasil yang didapatkan seperti dibawah ini :

   - E-book yang didapat lengkap dari link :

     http://www.nhk.or.jp/lesson/indonesian

   - Paper “VDVL Agile / Scrum” yang diterbitkan oleh *VDVL consultant telecom.*

- Artikel tentang Evidence-Based User Experience Research, Training, and Consulting yang dikumpulkan pada https://www.nngroup.com

3.3 Metode Pengembangan Sistem

Dalam pengembangan sebuah sistem diperlukan metode sebagai pedoman dalam merancang dan mengimplementasikannya pada sebuah sistem. Penelitian ini mengimplementasikan sebagian metode *Agile Scrum*. Pengembangan sistem ini akan dilakukan dua kali *sprint* dengan mempertimbangkan *backlog* produk yang ada, *sprint* pertama untuk menyelesaikan 80% *backlog* produk yang sudah dibuat, dan *sprint* kedua untuk menyempurnakan *product* yang sudah dibuat dan menyelesaikan 20% sisa *backlog* produk.

Tahap-tahap yang ada pada metode *Agile Scrum* yang diimplementasikan adalah :

1. Pembuatan *Backlog* Produk

   Backlog Produk adalah proses dimana peneliti mencatat seluruh fitur yang harus diimplementasikan saat proses pengembangan. Dalam hal ini peneliti menggunakan metode wawancara / berinteraksi langsung terhadap pengguna aplikasi. Yang akan didokumentasikan sebagai "*User Story*". Setiap *User Story* masing-masing akan diberi ID yang unik. Bentuk penulisan User Story akan dituliskan pada Tabel 3.3.

Tabel 0.3 Contoh User Story

ID	User Story
A-001	Sebagai pengguna harus dapat memilih jenis kata yang ingin dipelajari, Hiragana, Katakana, atau Kanji
A-002	Sebagai pengguna harus dapat memilih tingkatan dalam belajar, *Beginner*, *Intermediate,* atau *Advance*.
A-003	Dst.

   Dalam fase ini penulis membuat prioritas ke seluruh *User Story*, agar dapat memilah ketika ada perubahan kebutuhan, kita dapat mendahulukan *user story* yang terlebih dahulu kita rubah.

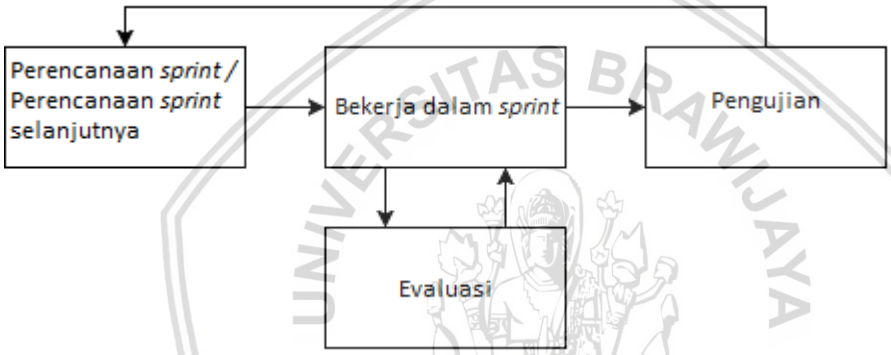
2. Perencanaan *Sprint* dan Pembuatan *Sprint Backlog*

   Penulis akan menentukan seberapa lama *sprint* yang akan dilakukan. Disini penulis mengambil 2 bulan penuh untuk melakukan *sprint*. Dengan perencanaan yang sebentar kita akan mengeluarkan beberapa banyak versi dalam waktu yang singkat, hasilnya adalah *feedback* dari pengguna akan kita terima lebih sering dan kemungkinan error dan bug akan sering juga.

Penulis akan menggunakan 2 sprint, maka *planning* yang dilakukan adalah menerapkan metode *incremental* dimana penulis akan mengerjakan bagian-bagian utama terlebih dahulu pada *sprint* pertama, kemudian akan dilanjutkan dengan membuat *planning* mengerjakan bagian yang ada perubahan dan menyempurnakan *backlog* produk yang belum dikerjakan.

Sebagai alternatif, kita diharapkan membuat *sprint* yang sedikit lebih lama. Ini juga akan memungkinkan *developer* bekerja lebih banyak dan tidak terburu-buru. Durasi yang penulis pilih untuk perencanaan *sprint* adalah 1 minggu, disini penulis bekerja sama dengan *stakeholders*. Sehingga penulis dapat menentukan prioritas *User Story* yang dibuat, sehingga *SCRUM* dapat diperkirakan berapa lama waktu yang dibutuhkan.

Setelah itu proses pengembangan dimulai berdasarkan prioritas. Dengan sistem ini, memungkinkan beberapa *User Story* dapat diselesaikan saat ditengah-tengah proses pengembangan, sehingga sangat memungkinkan untuk menyelesaikan semua *User Story* tepat waktu. Alur pembuatan aplikasi Hanasu digambarkan pada Gambar 3.2.

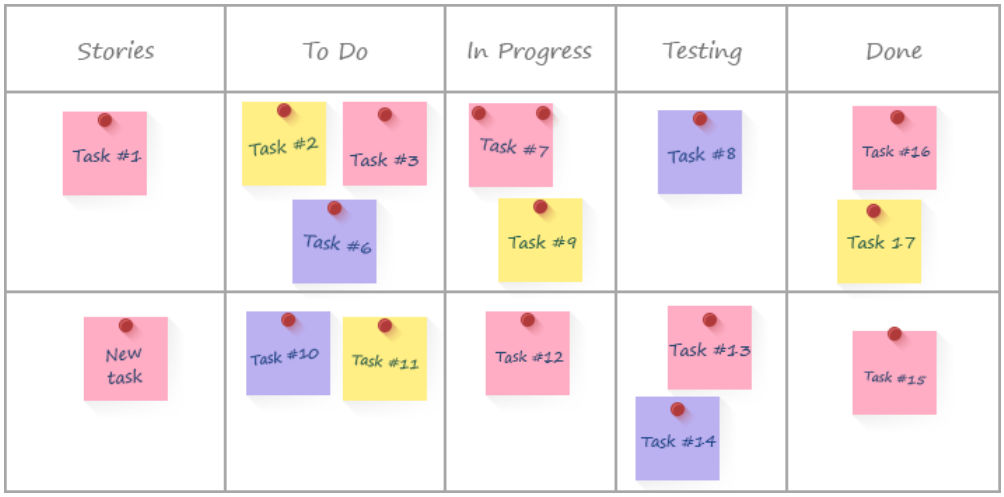


Gambar 0.2 Proses Scrum Agile Development

3. Bekerja dalam *Sprint* dan Evaluasi *SCRUM*

   Tahap ini adalah tahap yang penting dalam pengembangan sistem menggunakan metode scrum, penelitian ini mengganti proses *meeting* dengan evaluasi diri. Penulis bekerja mengevaluasi diri dengan menggunakan papan kerja, biasanya papan putih dengan dibagi menjadi 5 bagian "*Stories*", "*To do*", "*InProgress*", "*Testing*", dan "*Done*". Setiap pengerjaan dituliskan ke dalam *Sticky Note* berwarna dan ditempelkan ke setiap masing-masing fase. Note tersebut diurutkan berdasarkan prioritas.

   Ketika memulai proses pengerjaan suatu User Story, pindahkan sticker dari "*To Do*" ke "In *Progress*". Setelah selesai, pindahkan sticker menuju ke "*Testing*" dan setelah selesai di test dan tidak ada masalah, maka dipindahkan ke bagian "*Done*". Task board diilustrasikan seperti pada Gambar 3.3.



Gambar 0.3 Contoh Task Board

4. Pengujian dan Demonstrasi Produk

   Setelah melewati fase evaluasi *SCRUM* penulis mendemonstrasikan produk yang dibuat terhadap *Stakeholder*, setelah itu *Stakeholder* yang menentukan keputusan tentang proyek kedepannya apakah ada perubahan atau pergantian. Penelitian ini menggunakan *BlackBox Testing* untuk menguji fungsionalitas dari sistem dan menggunakan *Usability Testing* untuk menguji *usability system.*

   Pada *sprint* pertama, penelitian ini menguji hasil produk setelah benar-benar dapat diimplementasikan, kemudian mendapatkan kesimpulan dari hasil uji para *stakeholder*. Dilanjutkan dengan sprint kedua dimana produk yang dihasilkan sudah mengimplementasi semua *product backlog* dan sudah menyempurnakan bagian-bagian yang dirubah berdasarkan hasil uji pada *sprint* pertama. Pada masing-masing sprint saat pengujian akan selalu dilakukan *blackbox testing* untuk tetap menjaga fungsionalitas sistem.

5. Retrospektif dan perencanaan *sprint* selanjutnya

   Pada tahapan ini penulis mengadakan perencanaan untuk *sprint* selanjutnya, hal apa yang membuat lambat di *sprint* sebelumnya, yang dapat diperbaiki di *sprint* selanjutnya, sehingga mendapatkan proses yang lebih efisien lagi kedepannya. Perencanaan *sprint* selanjutnya dilakukan hanya pada *sprint* pertama karena penelitian ini hanya melakukan dua *sprint*.

3.4 Pembuatan *Backlog* Produk dan Pembuatan *Backlog Sprint*

Pada fase scrum Pembuatan *Backlog* Produk disini akan menjadi tahapan menganalisis kebutuhan sistem, dimana fase ini akan mendeskripsikan keseluruhan sistem. Pada fase Pembuatan *Sprint Backlog* dalam penelitian ini menentukan berapa lama setiap *sprint* yang akan dilakukan.

3.4.1 Analisis Kebutuhan

Didalam fase analisis sitem ini hal yang dilakukan pertama kali adalah mendeskripsikan sistem ke dalam gambaran umum agar mudah dipahami secara umum. Setelah itu menentukan lingkungan sistem yang akan dibuat. Kemudian mengidentifikasikan aktor yang akan menggunakan sistem ini secara rinci, setelah mengidentifikasikan aktor maka

dibuatkan *user story* berdasarkan cerita *user* yang sudah diwawancarai di tahap sebelumnya.

Selanjutnya membuat fungsionalitas berdasarkan *User Story* yang sudah dibuat. Fungsionalitas yang ditetapkan akan menjadi *backlog* produk yang nantinya akan menjadi kiblat dalam proses pengerjaan dari awal hingga akhir. Durasi *sprint* yang diambil adalah *shorter sprint*. Setelah mengetahui berapa lama *sprint* yang akan dilakukan perlu ada pembagian *backlog* produk dengan menggunakan metode *Split by Business Rules*.

Setelah itu penulisan fungsionalitas yang siap diimplementasi digambarkan menggunakan metode *Object Oriented Analysis* dengan menggunakan bahasa pemodel *UML (Unified Modeling Language)*. Dalam penggambaran perilaku aktor yang akan berinteraksi langsung dengan sistem akan menggunakan *Use Case* , di dalam sistem ini terdapat 1 *Use Case* dengan 1 jenis aktor. Setelah itu dibuatkan *Skenario Use Case* untuk lebih menjelaskan secara rinci setiap *Use Case.*

3.5 Bekerja dalam *Sprint and* Evaluasi Diri dalam *Scrum*

Dalam tahap ini diharapkan sudah mulai masuk ke dalam *sprint* yang artinya harus menyelesaikan segala *backlog* produk yang sudah ditetapkan pada subbab sebelumnya.

3.5.1 Perancangan

Setelah menentukan fungsionalitas, kemudian dilakukan perancangan diagram *class* untuk dapat mengetahui *class-class* apa saja yang dibutuhkan saat implementasi, dan dijelaskan setiap *class*-nya. Setelah merancang diagram *class* perlu dibuat diagram aktifitas untuk menggambarkan alur sistem dari sebuah *use case* yang sudah dibuat. Kemudian membuat diagram basis data untuk mengetahui data apa saja yang nantinya akan diolah.

Setelah itu dibuatlah sebuah perancangan algoritme untuk mempermudah saat pengerjaan implementasi. Barulah setelah itu membuat perancangan antarmuka beserta *screenflow* untuk membantu mengimplementasi antarmuka dan alur dari antarmuka tersebut.

3.5.2 Implementasi

Implementasi merupakan bagian dari fase Bekerja dalam *Sprint* dan Evaluasi dalam *SCRUM.* Perancangan perangkat lunak digunakan sebagai dasar acuan implementasi perangkat lunak yang akan dilakukan. Implementasi perangkat lunak diawali dengan penjabaran batasan implementasi perangkat lunak yang akan dikembangkan, dilanjutkan spesifikasi lingkungan pengembangan perangkat lunak. Kemudian dilanjutkan dengan implementasi basis data dengan menggunakan *Firebase*. Kemudian implementasi algoritme menjadi lanjutan dari tahap tersebut. Implementasi perangkat lunak dilakukan dengan menggunakan bahasa pemrograman Java dan *Google Speech-to-Text API Library*. Tahap terakhir dari implementasi adalah implementasi antarmuka berdasarkan perancangan yang telah dilakukan.

3.6 Pengujian dan Demonstrasi Produk

Pengujian dilakukan setiap setelah *sprint* berlangsung. Untuk mendapatkan hasil apakah ada *backlog* yang belum selesai atau sudah selesai semua. Apablia masih belum ada yang selesai maka akan dimasukkan ke dalam *backlog* produk pada *sprint* berikutnya.

3.6.1 Pengujian

Dalam tahap ini akan dilakukan pengujian terhadap sistem yang sudah dibuat apakah sistem sudah memenuhi spesifikasi kebutuhan yang melandasinya atau belum. Strategi yang dilakukan dalam pengujian ini adalah menggunakan *Blackbox Testing* untuk validasi keseluruhan sistem. Kemudian melakukan *Usability Testing* dengan menggunakan metode *A/B Testing* terhadap pengguna untuk mengetahui *Usabilty* sistem tersebut. Dalam Pengujian ini yang diuji adalah kecepatan dan kesesuaian tampilan yang ada. Di dalam fase ini penulis sudah masuk ke dalam fase pengujian dan demonstrasi produk dan perencanaan *Sprint* selanjutnya.

3.7 Retrospektif

Setelah selesai semua *sprint* maka dilakukan retrospektif atau melihat ulang kebelakang untuk mengetahui hasil atau kesimpulan dari semua yang telah dikerjakan.

3.7.1 Penarikan Kesimpulan

Pengambilan kesimpulan dilakukan setelah semua tahapan perancangan perangkat lunak, implementasi perangkat lunak, dan pengujian perangkat lunak telah selesai dilakukan. Kesimpulan diambil dari hasil pengujian dan analisis terhadap sistem yang dibangun. Tahap terakhir dari penulisan adalah saran yang dimaksudkan untuk memperbaiki kesalahan – kesalahan yang terjadi dan menyempurnakan penulisan serta untuk memberikan pertimbangan atas pengembangan perangkat lunak lebih lanjut.

BAB 4 ANALISIS KEBUTUHAN

Pada bab ini membahas tentang penelitian ini diharuskan mendapatkan *backlog* produk yang nantinya akan menjadi dasar untuk melakukan *sprint* hingga benar-benar menjadi produk jadi. Bab ini juga membahas tentang seberapa lama sprint yang akan dilakukan, bagaimana cara memecah *backlog* produk, dan bagaimana *user story* hasil dari wawancara.

4.1 Gambaran Umum Sistem

Sistem pembelajaran Bahasa Jepang yang menggunakan suara untuk belajar berbasis Android *Mobile.* Gambaran umum sistem dibagi menjadi 2 bagian, deskripsi sistem dan lingkungan sistem.

4.1.1. Deskripsi Umum Sistem

Dalam penelitian yang penulis ini dibangun sistem pembelajaran Bahasa Jepang menggunakan suara dalam bentuk *mobile* Android bernama Hanasu. Tujuan dari dikembangkannya sistem aplikasi Hanasu ini adalah untuk membiasakan pengguna untuk melatih dan menghafal kata-kata Bahasa Jepang lebih efisien. Pengguna akan diberi tampilan benda-benda dasar seperti buah-buahan, hewan, nama tempat, dan sebagainya, dan pengguna diminta menyebutkan dalam Bahasa Jepang. Terdapat juga mode kata-kata sehari-hari yang sekiranya sering diucapkan sehari-hari seperti ucapan selamat pagi, menyapa, menanyakan kabar, menanyakan letak, dan sebagainya, pengguna diminta untuk mengucapkan dalam Bahasa Jepang. Selain latihan terdapat juga kuis seperti ujian untuk menjawab beberapa soal agar dapat melatih pembelajaran.

4.1.2 Lingkungan Sistem

Aplikasi Hanasu ini dibangun menggunakan IDE Android Studio dengan bahasa pemrograman Java Android. Pada sisi basis data menggunakan basis data yang disediakan oleh google bernama *Firebase.*

4.2 Analisis Kebutuhan Perangkat Lunak

4.2.1 Identifikasi Aktor

Aktor adalah seseorang yang berinteraksi langsung dengan sistem. Adapun aktor yang berhubungan dengan sistem Hanasu dapat dilihat pada Tabel 4.1.

Tabel 0.1 Aktor Sistem

Aktor	Deskripsi
Pengguna	Merupakan orang yang langsung berinteraksi dengan aplikasi Hanasu dan ingin belajar Bahasa Jepang yang sudah dapat menggunakan seluruh fungsionalitas sistem.

4.2.2 Pembuatan *User Story*

Setelah melakukan wawancara terhadap calon pengguna sistem, penelitian ini membutuhkan *User Story* untuk menjadi calon fungsionalitas dari sebuah sistem. *User Story* adalah cerita dari cara pandang pengguna melihat sistem yang berbeda dengan cara pandang seorang *programmer*/*developer* melihat sistem. *User Story* dibuat berdasarkan jawaban dan saran dari narasumber. *User Story* dibuat hingga kedua belah pihak sepakat dan dapat dijadikan acuan untuk bekerja bersama-sama.

User Story dibuat menggunakan *INVEST mneomonic.* Dimana *INVEST* diciptakan untuk mengingatkan bagaimana cara membuat *User Story* yang baik (Bill Awake, 2011). Dimana *INVEST* sendiri merupakan akronim dari :

1. I – Independent : setiap *story item* harus berdiri sendiri dan tidak lebih dari 1 statement.
2. N – Negotiable : *story item* yang bagus merupakan *story* yang dapat dinegosiasi, *story* yang mementingkan esensi daripada detail.
3. V – Valuable : *story item* harus bernilai dan berguna untuk pengguna. Meski terkadang *developer* memiliki *concern* terhadap suatu hal, tapi mementingkan keperluan pengguna tetap harus diutamakan.
4. E – Estimable : *story item* yang baik adalah *story* yang dapat diestimasi waktu pengerjaannya sehingga dapat membantu pengguna untuk men-*ranking* kebutuhannya.
5. S – Small : *story item* yang bagus adalah *story* yang kecil dalam estimasi sudut pandang penggunaan dalam hitungan minggu bahkan hari. Bukan *story* yang memandang "ini akan memakan waktu berbulan-bulan".
6. T – Testable : *story item* yang bagus tentunya merupakan *story* yang diperkirakan dapat dieksekusi dan dapat diuji meski kita belum memulai mengerjakannya.

Berdasarkan hasil wawancara yang dilakukan didapatkanlah sebuah *User Story*. Dalam penulisannya penelitian ini menggunakan format sederhana : *Sebagai <tipe pengguna >, saya ingin <tujuan> sehingga saya <alasan>*

Sehingga didapatlah *User Story* yang dapat dilihat pada Tabel 4.2.

Tabel 0.2 *User Story* Hasil Wawancara

No	*User Story*
1	Sebagai pengguna saya ingin dapat menjawab soal menggunakan suara untuk melatih kemampuan *speaking* saya dalam Bahasa Jepang
2	Sebagai pengguna saya ingin mengetahui apakah cara bicara saya sudah benar atau belum untuk evaluasi diri.
3	Sebagai pengguna saya ingin mendapatkan bantuan untuk menggunakan sistem agar lebih mudah menggunakannya.
4	Sebagai pengguna saya ingin mendapat contoh bagaimana cara penyampaian yang benar dalam Bahasa Jepang supaya saya dapat melatih kemampuan *listening* saya.

5	Sebagai pengguna saya ingin belajar mulai dari huruf-huruf Jepang karena saya benar-benar baru belajar Bahasa Jepang.
6	Sebagai pengguna saya ingin belajar kosa kata Bahasa Jepang supaya kosa kata berbahasa Jepang saya bertambah.
7	Sebagai pengguna saya ingin ditampilkan gambar-gambar supaya tidak bosan dalam belajar.
8	Sebagai pengguna saya ingin dapat login di *smartphone* apapun ketika ingin berpindah perangkat data saya tidak hilang dan harus belajar memulai dari awal.

4.2.3 Kebutuhan Fungsional Sistem

Kebutuhan fungsional sistem merupakan fungsi di dalam sistem atau komponen yang harus ada (ISO/IEC/IEEE 24765, 2010). Fungsionaliitas ini didapat dari *User Story* yang sudah dibuat pada bab sebelumnya, yang nantinya akan menjadi *backlog* produk dalam sistem. Di dalam penulisannya harus diberi suatu identitas untuk mempermudah proses identifikasi kebutuhan juga menjaga konsistensi terkait dengan kebutuhan suatu sistem, mulai dari proses perancangan hingga pengujian selesai dilakukan. *Backlog* Produk sistem serta spesifikasinya dapat dilihat pada Tabel 4.3.

Tabel 0.3 Kebutuhan Fungsionalitas Sistem

No	Kode Kebutuhan Sistem	Deskripsi Kebutuhan
1	HNS-001	Sistem harus mampu menyediakan tempat untuk dapat masuk ke dalam sistem menggunakan akun google.
2	HNS-002	Sistem harus mampu menyediakan mekanisme untuk menjawab soal menggunakan suara.
3	HNS-003	Sistem harus mampu menyediakan mekanisme penilaian benar atau salah.
4	HNS-004	Sistem harus mampu menyediakan fungsi bantuan untuk mempermudah pengguna.
5	HNS-005	Sistem harus mampu menyediakan mekanisme mengeluarkan suara apabila menekan salah satu materi.
6	HNS-006	Sistem harus mampu menyediakan materi tentang huruf Bahasa Jepang.

7	HNS-007	Sistem harus mampu menyediakan materi tentang huruf kosakata Bahasa Jepang.
8	HNS-008	Sistem harus mampu menyediakan soal berupa gambar.

4.2.4 Kebutuhan Non-Fungsionalitas

Kebutuhan non-fungsionalitas adalah kebutuhan yang tidak ter-*cover* pada kebutuhan fungsionalitas. Kebutuhan yang berisikan batasan-batasan dalam pelayanan sistem atau sebuah fungsi yang ditawarkan oleh sistem tersebut seperti Skalabilitas, Kapasitas, Ketersediaan, Keamanan, dan banyak lagi (Ian Sommerville, 2004).

Kebutuhan non-fungsionalitas dari aplikasi ini hanya meliputi *Usability* dengan parameter *Learnability* dan *Satisfaction*. Sebuah antarmuka harus dapat mudah digunakan dan nyaman untuk digunakan (Jakob Nielsen, 2012).

4.2.5 Perencanaan Durasi *Sprint*

Perencanaan durasi setiap *sprint* akan menentukan seberapa lama *sprint* yang akan dilakukan. Dalam metode *Agile Scrum* setiap *sprint* harus berorientasikan terhadap waktu bukan kebutuhan fungsionalitas, *sprint* yang dilakukan harus memiliki periode waktu yang tetap, antara 1 – 4 minggu (Mark Levison, 2013). Jenis sprint yang terdapat dalam sprint dibagi 2 yakni *"Longer Sprint"* dan "*Shorter Sprint*". Dimana *Longer Sprint* berdurasi 3 – 4 minggu dan *Shorter Sprint* berdurasi 1 – 2 minggu.

Penelitian ini menggunakan *Shorter Sprint* karena memiliki banyak kelebihan dan sedikit kekurangan dibandingkan *Longer Sprint.* Keuntungan *Shorter Sprint* diantaranya :

- Disaat waktu sprint yang sebentar maka banyak memiliki kesempatan untuk mencoba beberapa perubahan kecil.
- Dapat menyediakan waktu lebih untuk belajar.
- *Cycle* yang lebih pendek lebih mudah untuk dibuat *planning*-nya dan juga meningkatkan fokus.

4.2.6 Perencanaan Pembagian *Backlog* Produk

Fase perencanaan pembagian *product backlog* atau yang disebut *Sprint Backlog Creation* merupakan fase untuk memecah dari sejumlah *backlog* produk yang dibuat untuk dibagi menentukan berapa *backlog* produk yang akan dikerjakan di setiap *sprint*-nya. Ada beberapa metode untuk memecah *backlog* produk menurut Christiaan Verwijs *:*

1. *Split by Workflow Steps*

   Memecah *backlog* produk berdasarkan alur kerja yang sudah ada, sehingga memecah *backlog* produk berdasarkan urutan yang sudah dibuat dan memecah rata sejumlah *sprint* yang akan dilakukan.

2. *Split by Business Rules*

   Memecah *backlog* produk berdasarkan aturan bisnis yang sudah dibuat berdasarkan *user story*. Biasanya pembagian seperti ini dilakukan apabila sistem yang akan dibuat memiliki fitur yang perlu dikerjakan terlebih dahulu karena pengaruh proses bisnis mereka. Maka kita mendahulukan *backlog* produk yang bersifat mendesak.

3. *Split by Happy / Unhappy Flow*

   Memecah *backlog* produk berdasarkan bagian yang menurut pengguna menyenangkan atau tidak menyenangkan. Mendahulukan proses yang tidak menyenangkan terlebih dahulu agar dapat mengetahui secara spesifik bagian mana yang penting terlebih dahulu. Bagian tidak menyenangkan dapat digambarkan dalam bentuk ketakutan akan terjadinya hal yang tidak diinginkan.

4. *Split by Input Options / Platform*

   Memecah *backlog* produk dengan mempertimbangkan beberapa *platform* yang akan menjalankan sistem ini. Dengan ini pengembang dapat membagi *sprint* berdasarkan *platform* yang paling susah untuk diimplementasi seperti PC.

5. *Split by Datatypes or Parameters*

   Memecah *backlog* produk dengan mempertimbangkan parameter yang ingin dikerjakan terlebih dahulu, apabila pengembang memiliki parameter agar sistem dapat menampilkan data terlebih dahulu, maka *sprint* pertama yang dilakukan adalah menampilkan data dalam bentuk dasar atau tabel. *Sprint* berikutnya adalah menampilkan data dalam bentuk diagram, grafik, atau sebagainya.

6. *Split by Operations*

   Memecah *backlog* produk berdasarkan *CRUD (Create Read Update Delete).* Sehingga kita bisa menentukan prioritas dari hal tersebut.

7. *Split by Roles*

   Memecah *backlog* produk berdasarkan *role* yang ada dalam proses model bisnis. Kita dapat menentukan prioritas yang diambil berdasarkan *role* mana yang lebih penting untuk dikerjakan terlebih dahulu, semisal apakah bagian admin atau pengguna biasa terlebih dahulu.

8. *Split by Test Case*

   Memecah *backlog* produk dengan memilih bagian *backlog* produk mana yang memiliki *test case* yang tidak beresiko tinggi akan dikerjakan terlebih dahulu.

Penelitian ini menggunakan *Split by Business Rules* dimana menentukan inti dari sistem yang akan dikerjakan terlebih dahulu. Sehingga didapatkan pembagian *sprint* seperti dibawah ini :

Sprint pertama didapatkan pada Tabel 4.4 yang menjadi prioritas pada *backlog* produk.

Tabel 0.4 *Backlog* Produk untuk Sprint Pertama

No	Kode Kebutuhan Sistem	Deskripsi Kebutuhan
1	HNS-002	Sistem harus mampu menyediakan mekanisme untuk menjawab soal menggunakan suara.
2	HNS-003	Sistem harus mampu menyediakan mekanisme penilaian benar atau salah.
3	HNS-004	Sistem harus mampu menyediakan fungsi bantuan untuk mempermudah pengguna.
4	HNS-005	Sistem harus mampu menyediakan mekanisme mengeluarkan suara apabila menekan salah satu materi.
5	HNS-006	Sistem harus mampu menyediakan materi tentang huruf dalam Bahasa Jepang.
6	HNS-007	Sistem harus mampu menyediakan materi tentang kosa-kata dalam Bahasa Jepang.

Sprint kedua ditunjukkan pada tabel 4.4 dibawah ini yang merupakan bukan prioritas dari *backlog*:

No	Kode Kebutuhan Sistem	Deskripsi Kebutuhan
1	HNS-001	Sistem harus mampu menyediakan tempat untuk dapat masuk ke dalam sistem menggunakan akun google.
2	HNS-008	Sistem harus mampu menyediakan soal berupa gambar.

4.2.7 Perancangan *Use Case* Digram

Pemodelan *Use Case* merupakan pemodelan yang sering digunakan di dalam pengembangan perangkat lunak sebagai pendekatan untuk mendeskripsikan kebutuhan perangkat lunak (Ivar Jacobson , 1992). Dari pemodelan *Use Case* ini memungkinkan kita untuk menentukan representasi lain dari sebuah sistem, bagaimana cara aktor berinteraksi langsung dengan sistem. Perancangan *Use Case* dari aplikasi Hanasu digambarkan pada Gambar 4.1.

Penjelasan dari setiap *Use Case* dijelaskan diantaranya :

Pada *Use Case* Masuk menggunakan akun google, aktor hanya dapat menggunakan seluruh fitur aplikasi Hanasu apabila sudah masuk ke dalam sistem menggunakan akun google.

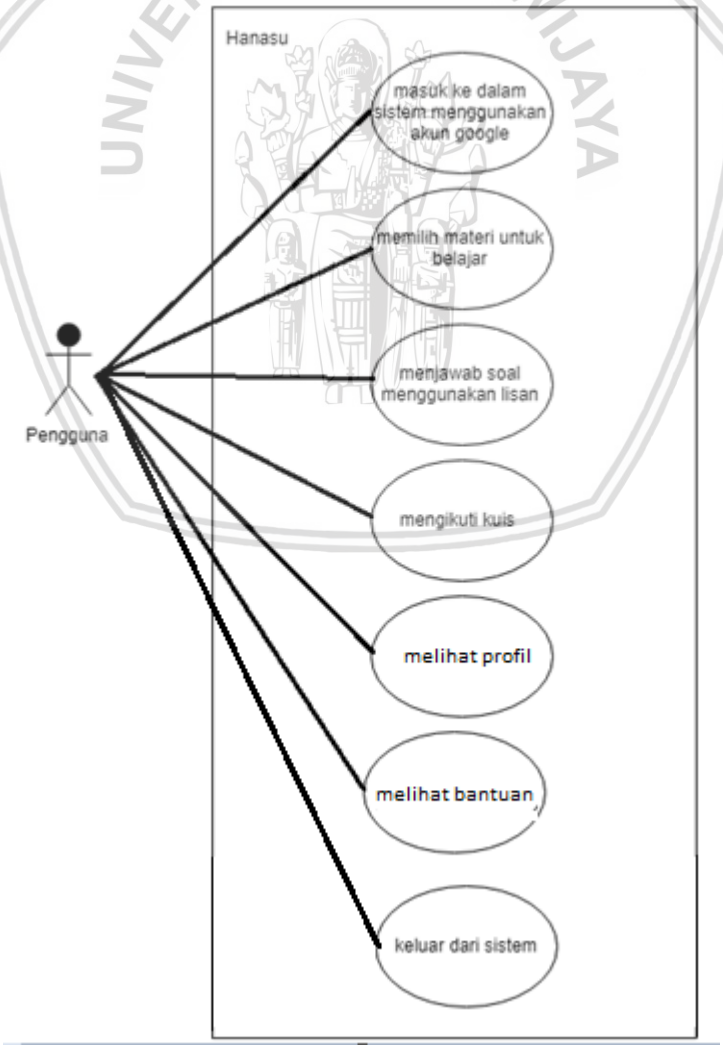
Pada *Use Case* Melihat materi untuk belajar, aktor dapat melihat list – list dari seluruh huruf Hiragana dan Katakana atau kosa kata Bahasa Jepang, dimana ketika aktor menekan salah satu materi , sistem akan mengeluarkan suara sesuai huruf yang ditekan.

Pada *Use Case* Menjawab soal menggunakan lisan, aktor akan dihadapkan dengan beberapa gambar atau soal, dimana menjawab soal tersebut menggunakan lisan.

Pada *Use Case* Mengikuti kuis, aktor hanya dapat mengambil kuis apabila sudah masuk ke dalam sistem, kuis yang diambil akan mendapat nilai dari tiap materi yang diambil.

Pada *Use Case* Melihat profil, aktor dapat melihat profilnya berupa nama, email, dan materi yang sudah pernah dijawab.

Pada *Use Case* Melihat bantuan, aktor hanya dapat melihat bantuan ketika pertama kali setelah menginstall aplikasi.



Gambar 0.1 Use Case Sistem Hanasu

4.2.8 Skenario *Use Case*

Skenario *Use Case* melihat materi untuk belajar dapat dilihat pada Tabel 4.5.

Tabel 0.5 *Skenario Use Case* Melihat materi untuk belajar

Use Case Name	Melihat materi untuk belajar
Actor	Pengguna
Target	Menampilkan materi untuk dipelajari berdasar visual dan lisan
Pre Condition	Pengguna harus sudah terautentifikasi masuk ke dalam sistem
Main Flow	1. Memilih salah satu materi 2. Menekan tombol untuk mendengar cara pengucapan yang benar dari sebuah huruf.
Post Condition	1. Mengetahui tulisan huruf hiragana / katakana atau kosa kata dalam bahasa Jepang 2. Mengetahui cara pengucapan yang benar dari huruf hiragana / katakana atau kosa kata dalam Bahasa Jepang
Alternative Flow	-

Skenario *Use Case* masuk menggunakan akun google dapat dilihat pada Tabel 4.6.

Tabel 0.6 *Skenario Use Case* Masuk menggunakan akun google

Use Case Name	Masuk menggunakan akun google
Actor	Pengguna
Target	Terautentifikasi sebagai pengguna aktif
Pre Condition	-
Main Flow	1. Masuk menggunakan akun google 2. Konfirmasi yang sudah disediakan oleh google
Post Condition	1. Bisa mengolah data profil 2. Bisa mendapatkan soal quiz untuk menambah level
Alternative Flow	-

Skenario *Use Case* menjawab soal menggunakan lisan dapat dilihat pada Tabel 4.7.

Tabel 0.7 *Skenario Use Case* Menjawab soal menggunakan lisan

Use Case Name	Menjawab soal menggunakan lisan
Actor	Pengguna
Target	Menjawab soal dengan suara dari pengguna

Pre Condition	Pengguna harus sudah terautentifikasi masuk ke dalam sistem
Main Flow	1. Memilih salah satu soal 2. Menekan tombol untuk memulai berbicara untuk menjawab soal 3. Mendapat respon apakah pengucapan sudah benar atau belum
Post Condition	1. Mengetahui cara pengucapan yang benar 2. Mendapat point
Alternative Flow	-

Skenario *Use Case* mengikuti kuis dapat dilihat pada Tabel 4.8.

Tabel 0.8 *Skenario Use Case* Mengikuti kuis

Use Case Name	Mengikuti kuis
Actor	Pengguna
Target	Mendapatkan soal-soal kemudian dijawab dan mendapat point
Pre Condition	Pengguna harus sudah terautentifikasi masuk ke dalam sistem
Main Flow	1. Membuka soal 2. Menjawab soal menggunakan lisan dengan waktu yang sudah ditentukan. 3. Mendapat hasil akhir dari beberapa soal yang sudah dikerjakan.
Post Condition	1. Mendapat point berdasarkan jumlah soal yang berhasil dijawab.
Alternative Flow	-

Skenario *Use Case* melihat profil dapat dilihat pada Tabel 4.9.

Tabel 0.9 *Skenario Use Case* Melihat profil

Use Case Name	Melihat profil
Actor	Pengguna
Target	Menampilkan profil
Pre Condition	Pengguna harus sudah terautentifikasi masuk ke dalam sistem
Main Flow	1. Melihat status profil saat ini berupa nama, tanggal lahir, level, point, dan avatar.
Post Condition	1. Menampilkan profil data diri.

Alternative Flow	-

Skenario *Use Case* melihat materi untuk belajar dapat dilihat pada Tabel 4.10.

Tabel 0.10 Skenario *Use Case* Melihat Bantuan

Use Case Name	Melihat Bantuan
Actor	Pengguna
Target	Mendapat bantuan penggunaan
Pre Condition	Baru pertama kali membuka aplikasi setelah meng-*install*
Main Flow	1. Membuka aplikasi saat pertama kali 2. Mendapat bantuan berupa langkah-langkah penggunaan
Post Condition	1. Mendapatkan bantuan
Alternative Flow	-

Skenario *Use Case* keluar dari sistem dapat dilihat pada Tabel 4.11.

Tabel 0.11 *Skenario Use Case* Keluar dari sistem

Use Case Name	Keluar dari sistem
Actor	Pengguna
Target	Keluar dari sistem
Pre Condition	Pengguna harus sudah terautentifikasi masuk ke dalam sistem
Main Flow	1. Keluar dari sistem 2. Kembali menuju halaman login
Post Condition	1. Tidak terautentifikasi
Alternative Flow	-

BAB 5 PERANCANGAN DAN IMPLEMENTASI

Perancangan dilakukan untuk membuat gambaran sebelum implementasi yang akan dibagi dalam beberapa tahap, pertama membuat Diagram *Class* , Setelah itu membuat Diagram Aktifitas, dilanjutkan dengan merancang Diagram Basis Data, setelah itu merancang Diagram Algoritme, dan diakhiri dengan merancang Design Antar Muka.

5.1 Diagram *Class*

Diagram *Class* merupakan diagram yang menggambarkan pemodelan c*lass-class* yang digunakan untuk merancang detail dari elemen-elemen *Use Case* yang sudah dimodelkan. Dalam perancangan Diagram *Class* juga akan dibagi 2 *sprint*, karena saat terdapat perubahan dalam sistem juga memungkinkan terdapat perubahan pada Diagram *Class* tersebut.

5.1.1 Sprint Pertama

Sprint pertama dimana penelitian ini fokus terhadap 6 dari 8 *backlog* produk yang akan diselesaikan. Dimana Diagram *Class* pada *sprint* pertama dapat dilihat pada Gambar 5.1.

Diagram *Class* pada Gambar 5.1 merupakan Diagram *Class* yang sudah siap diimplementasi. Terdapat beberapa bagian *Class* seperti *Activity*, *Fragment*, *Model*, dan *Controller*.

- **Activity**
  CharacterActivity, class yang berisikan tentang seluruh huruf yang ada dalam Bahasa Jepang, terdiri dari *TextToSPeech*, *Toolbar*, *TextView*.

  CourseActivity, class yang berisikan tentang materi yang akan dipelajari, berisikan tentang *RecyclerView*, *Toolbar*, *TextView*, *TextToSpeech*, dan *String array* “nihongo”, “arti”, “latin”, dan “deskripsi”.

  MainActivity, class yang berisikan *fragment* ditampung di dalam *view pager*, berisikan *ViewPager* dan *BottomNavigationView*.

  QuizActivity, *class* yang berfungsi untuk kuis berisikan tentang *Toolbar* dan *PrgoressBar*.

- **Fragment**

  QuizFragment, merupakan tampilan dari halaman jenis-jenis kuis terdiri dari *RecyclerView*.

  BelajarFragment, merupakan tampilan dari halaman jenis-jenis Belajar terdiri dari *RecyclerView.*

- **Model**
  ModelCourse, berisikan tentang data title *String*, *image Int*, deskripsi *String*, dan *progress Int*.

  ModelLesson, berisikan tentang data *nihongo String*, arti *String*, latin *String*, dan deskripsi *String*.

  ModelQuiz, berisikan tentang data *title String* dan deskripsi *String*.

- **Controller dan Adapter**

CourseAdapter, menampung seluruh data dan meng-*handle course* berisi *List ModelCourse* dan *Context*.

LessonAdapter, menampung seluruh data dan meng-*handle Lesson* berisi *List ModelLesson* dan *Context*.

QuizAdapter, menampung seluruh data dan meng-*handle* kuiss berisi *List mQuiz* dan *Context*.

CharacterController, mengambli data dari *CharacterModel* yang kemudian akan dikembalikan menuju *CharacterActivity.*

CourseController, mengambli data dari *CourseModel* ,menampung jawaban dan hasil suara dari yang diomongkan oleh pengguna, kemudian dimunculkan apakah benar atau salah.

ViewPagerAdapter, merupakan *adapter* yang berfungsi untuk menjadi jembatan antara *MainActivity* dengan *QuizFragment,* dan *LessonFragment.*

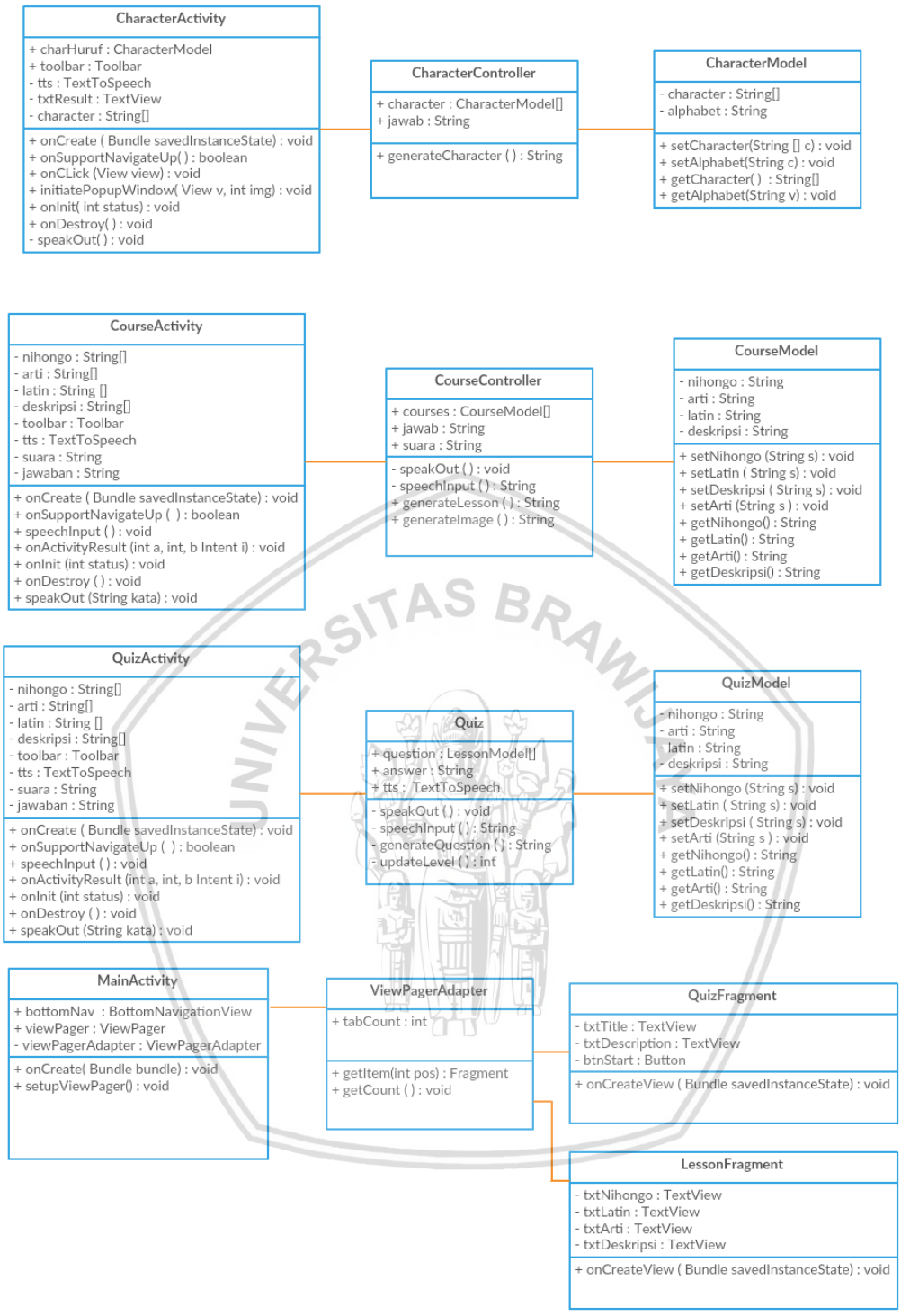
5.1.2 Sprint Kedua

Pada *sprint* kedua Diagram *Class* terjadi sedikit perubahan pada *ProfileFragment* dan *ModelProfile.* Diagram *Class* pada sprint kedua dapat dilihat pada Gambar 5.2 dibawah ini :

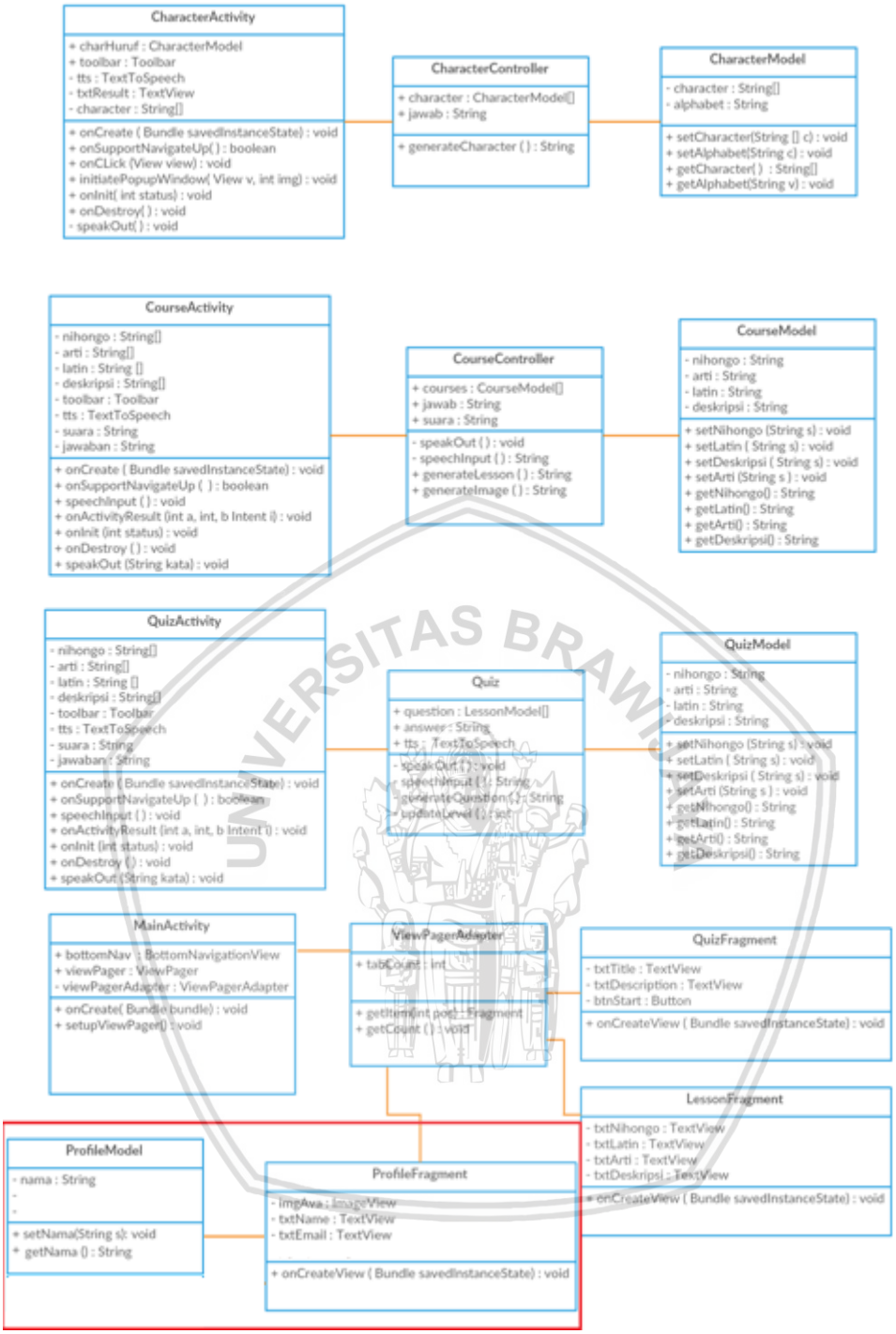
Dengan perubahan ditandai dengan kotak berwarna merah, dan masing-masing keterangan *class* seperti dibawah ini :

ProfileFragment, merupakan tampilan untuk halaman profil yang tertempel di halaman *MainActivity*. Terdiri dari *ImageView* dan 2 *TextView*.

ModelProfile, berisikan tentang data *title String* dan deskripsi *String*.



Gambar 0.1 Diagram *Class* Hanasu *Sprint* Pertama



Gambar 0.2 Diagram *Class* Hanasu Sprint Kedua

5.2 Diagram Aktifitas

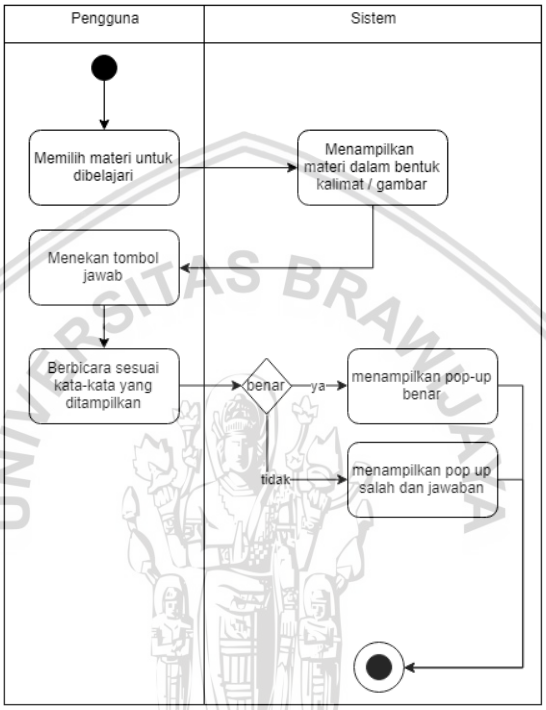
Diagram ini menjelaskan aspek dinamis dari sebuah sistem, yang mana menjelaskan aliran aktifitas yang terjadi antara aktor dengan sistem yang tengah dikembangkan. Diagram aktifitas yang akan dibuat oleh penulis diantaranya, diagram

aktifitas belajar materi, diagram aktifitas mengikuti kuis, diagram aktifitas masuk ke dalam sistem. Diagram aktifitas dibuat menjadi 2 *sprint* :

5.2.1 Sprint Pertama

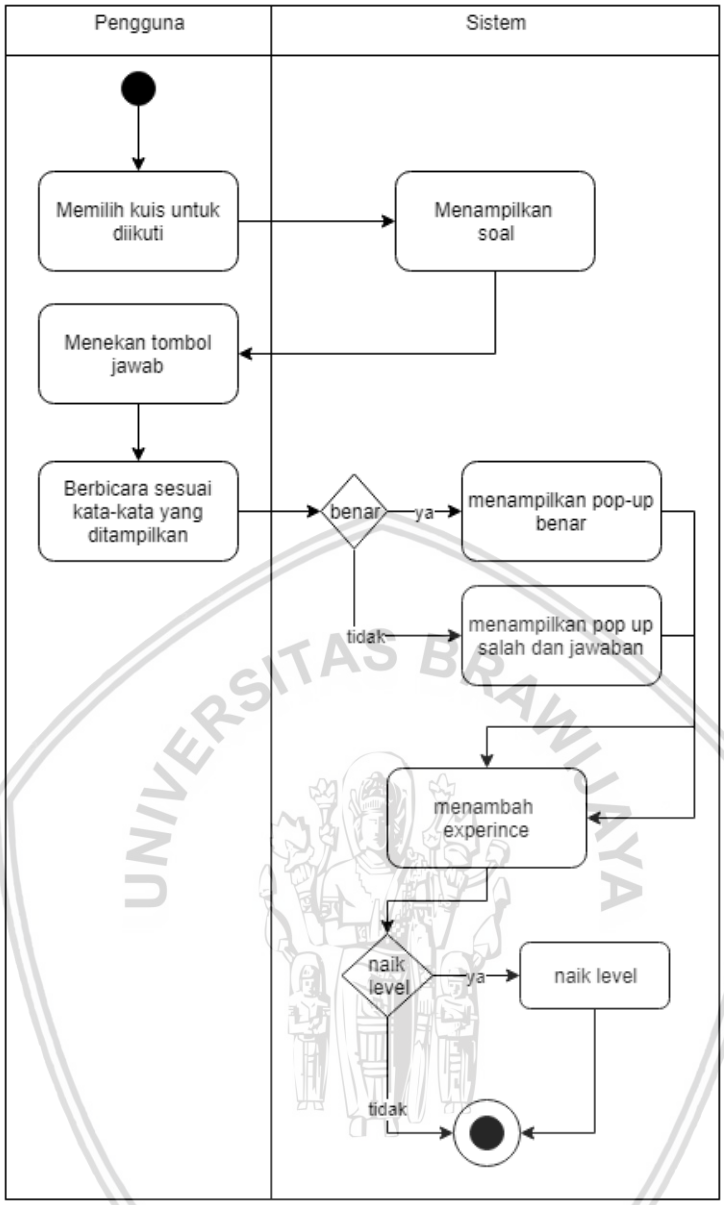
Sprint pertama penelitian ini mengambil 2 contoh dari beberapa *Use Case*, diantaranya *Use Case* Menjawab Soal Menggunkan Lisan yang digambarkan menjadi diagram aktifitas belajar materi dan *Use Case* Mengikuti Kuis yang digambarkan pada diagram aktifitas mengikuti kuis.

Diagram aktifitas belajar materi dapat dilihat Gambar 5.3.



Gambar 0.3 Diagram Aktifitas Belajar Materi

Diagram aktifitas mengikuti kuis dapat dilihat pada Gambar 5.4.

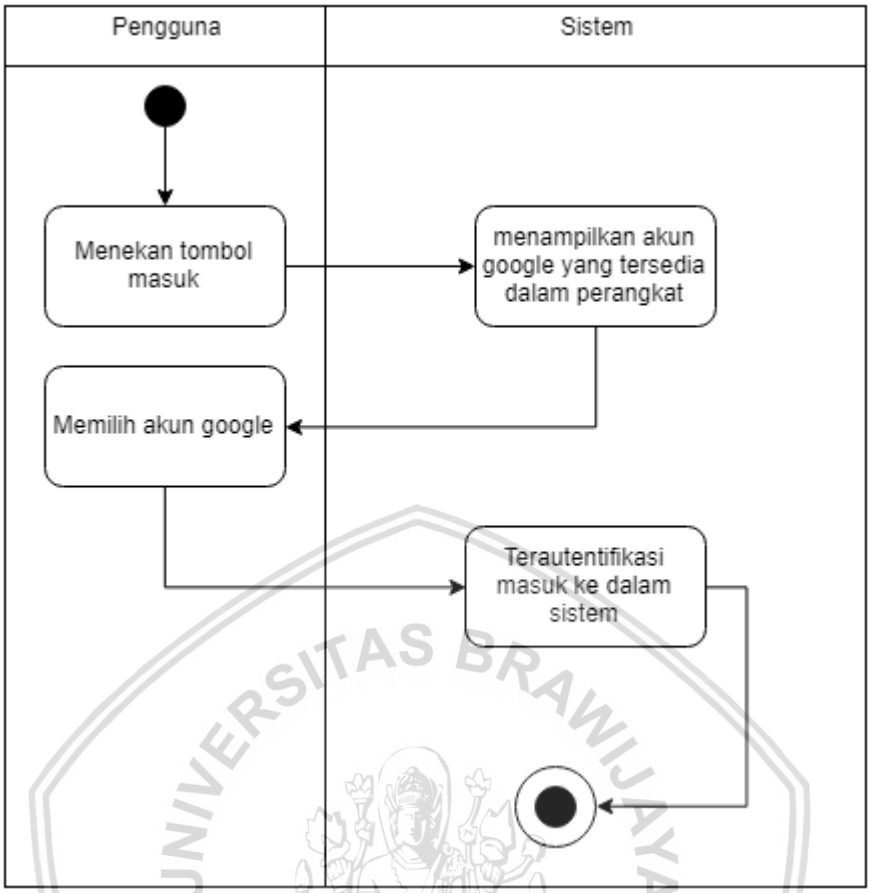


Gambar 0.4 Diagram Aktifitas Mengikuti Kuis

5.2.2 Sprint Kedua

Sprint kedua mengambil 1 contoh dari *Use Case* masuk menggunakan akun google yang digambarkan pada diagram aktifitas masuk ke dalam sistem.

Diagram aktifitas untuk masuk ke dalam sistem dapat dilihat pada Gambar 5.5.



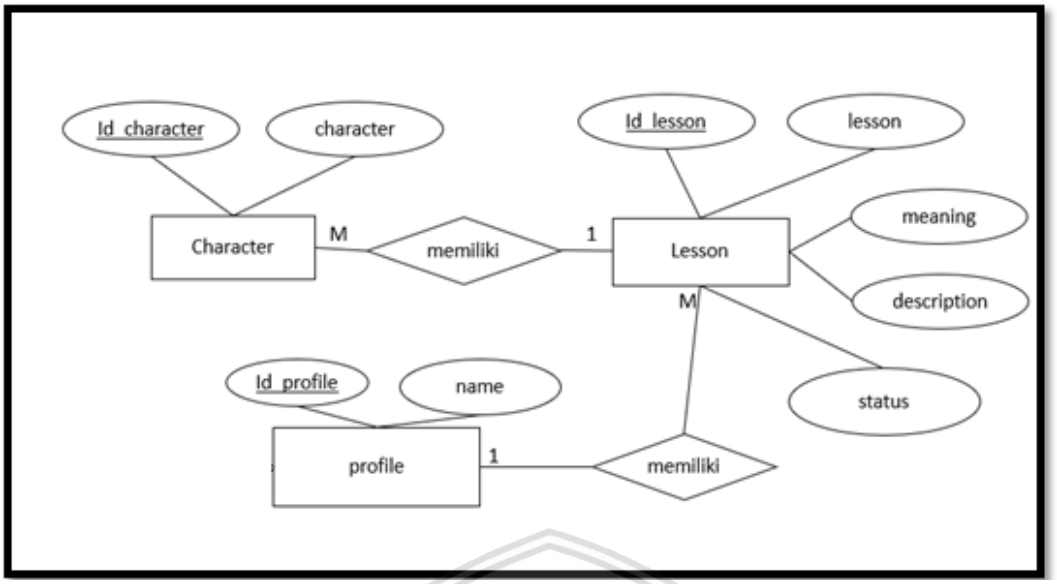
Gambar 0.5 Diagram Aktifitas Masuk Ke Dalam Sistem

5.3 Perancangan Basis Data

Basis data merupakan tempat penyimpanan data yang dibutuhkan oleh sistem. Perancangan basis data diperlukan untuk mengkontrol data yang masuk dan keluar dari sistem agar lebih tertata. Perancangan basis data diambil berdasarkan analisis kebutuhan yang dilakukan di tahap awal dari perancangan sistem. Perancangan basis data akan dibagi menjadi 2 *sprint*.

5.3.1 Sprint Pertama

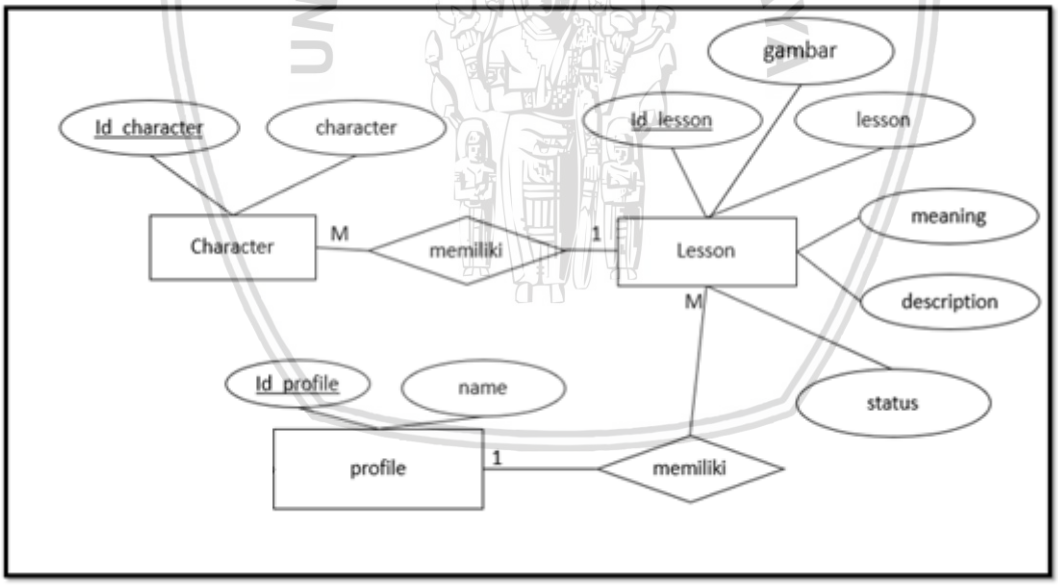
Perancangan database pada *sprint* pertama didapatkan perancangan basis data seperti Gambar 5.6.



Gambar 0.6 ERD Hanasu *Sprint* Pertama

5.3.2 Sprint Kedua

Perancangan basis data pada *sprint* kedua terdapat sedikit penambahan pada basis data *lesson* yang memiliki atribute baru bernama "gambar". Perubahan dapat dilihat pada Gambar 5.7.



Gambar 0.7 ERD Hanasu *Sprint* Kedua

5.4 Perancangan Algoritme

Setelah mendapatkan data yang akan diolah dan diagram aktifitas yang sudah dibuat, maka langkah selanjutnya adalah merancang algoritme yang digambarkan dalam bentuk *Pseudocode*. Fungsi dari perancangan algoritme disini agar dapat mengetahui cara menyelesaikan masalah secara tepat dan efisien. Perancangan algoritme juga dibagi menjadi dua *sprint*, *sprint* pertama akan membahas tentang 3 algoritme , algoritme belajar menggunakan lisan, algoritme mengikuti kuis, dan

algoritme masuk ke dalam sistem. Perancangan algoritme juga akan dibagi menjadi 2 sprint.

5.4.1 Sprint Pertama

Algoritme belajar menggunakan lisan dapat dilihat pada Tabel 5.1

Tabel 0.1 *Pseudocode* Algoritme Belajar Menggunakan Lisan

```
1   Function speechInput()
2       tts -> TextToSpeech
3       soal -> String
4       tts.speak()
5       if(tts != null)
6           if(checked(tts.result(), soal)
7                alert("benar")
8           else
9               alert("salah, jawaban anda " + tts.result())
10          endif
11      endif
12  end function
13
14  function bool checked(answer -> String, compare -> String)
15      wrong -> int
16      loop (i -> 0, i < answer.length, i++)
17          if(answer.at[i] != compare.at[i])
18              wrong++
19          endif
20      endloop
21      if(wrong > 0)
22          return false
23      else
24          return true
25      endif
26  end function
27
```

Algoritme di belajar menggunakan lisan merupakan algoritme untuk mengecek apakah kalimat yang diucapkan sudah benar atau tidak dengan cara mengecek setiap huruf yang diucapkan dengan huruf yang sebenarnya. Untuk Algoritme mengikuti kuis dapat dilihat pada algoritme Tabel 5.2.

Tabel 0.2 *Pseudocode* Algoritme Mengikuti Kuis

```
1   Profile -> Profile
2   Total -> 0
3   Soals -> Soal[]
4   Function speechInput()
5       tts -> TextToSpeech
6       soal -> String
7       tts.speak()
8       if(tts != null)
9           if(checked(tts.result(), soal)
10               alert("benar")
11               total ++
12          else
13              alert("salah, jawaban anda " + tts.result())
14          endif
15      endif
16  end function
```

```
17
18  function checked(answer -> String, compare -> String)
19      wrong -> int
20      loop (i -> 0, i < answer.length, i++)
21          if(answer.at[i] != compare.at[i])
22              wrong++
23          endif
24      endloop
25      if(wrong > 0)
26          return false
27      else
28          return true
29      endif
30  end function
31
32  function expUp ()
33      if(total * (100 / soal.size) > expNow)
34          course.exp = total * (100/ soal.size)
35      endif
36  end function
```

Algoritme mengikuti kuis merupakan algoritme untuk mengikuti kuis dan menjelaskan bagaimana *experience* bekerja. Apabila soal terjawab dengan benar lebih dari jumlah jawaban sebelumnya, maka exp akan berubah menjadi exp saat ini yang diperoleh. Untuk Algoritme masuk ke dalam sistem dijelaskan pada Tabel 5.3.

Tabel 0.3 *Pseudocode* Algoritme Masuk ke Dalam Sistem

```
1   googleAcc -> GoogleAuth
2   function signin()
3       open google auth
4       if(google account exist)
5           choose google account
6           go to main page
7       endif
8   end function
9
10  function signout()
11      google auth sign out
12      go to login page
13  end function
```

Algoritme masuk ke dalam sistem merupakan algoritme untuk masuk ke dalam sistem menggunakan akun *Google*.

5.4.2 Sprint Kedua

Pada sprint kedua terdapat 1 perubahan algoritme pada belajar menggunakan lisan, apakah soal berupa gambar atau bukan. Sehingga terjadi perubahan algoritme yang dapat dilihat pada Tabel 5.4.

Tabel 0.4 *Pseudocode* Algoritme Belajar Menggunakan Lisan

```
1   Soals -> Soal[]
2   Function speechInput()
3       tts -> TextToSpeech
```

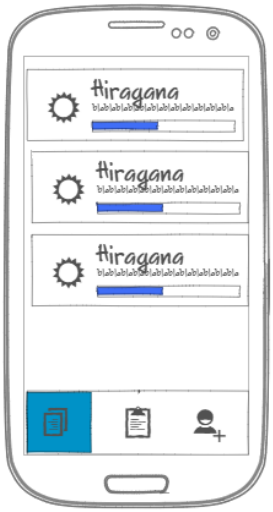
```
4       soal -> String
5       tts.speak()
6       if(tts != null)
7           if(soals.type == gambar)
8               show gambar
9           endif
10          if(checked(tts.result(), soal)
11               alert("benar")
12          else
13              alert("salah, jawaban anda " + tts.result())
14          endif
15      endif
16  end function
17
18  function bool checked(answer -> String, compare -> String)
19      wrong -> int
20      loop (i -> 0, i < answer.length, i++)
21          if(answer.at[i] != compare.at[i])
22              wrong++
23          endif
24      endloop
25      if(wrong > 0)
26          return false
27      else
            return true
        endif
    end function
```

5.5 Perancangan Antarmuka

Perancangan antarmuka dibutuhkan untuk mempermudah mendapatkan gambaran desain saat implementasi. Perancangan antarmuka menggunakan *tools* "NinjaMock" yang dapat merancang desain secara online di web tanpa perlu memasang aplikasi tertentu terlebih dahulu. Perancangan antarmuka disesuaikan dengan kebutuhan yang sudah disebutkan pada subbab sebelumnya. Perancangan antarmuka juga dibagi menjadi 2 *sprint*.

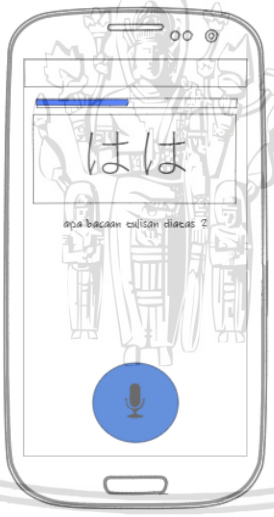
5.5.1 Sprint Pertama

Pada *Sprint* pertama terdapat 5 perancangan antarmuka diantaranya antarmuka halaman memilih materi, antarmuka halaman menjawab soal, antarmuka profil, antarmuka memilih kuis, dan antar muka *splashscreen*. Antarmuka memilih materi dapat dilihat pada Gambar 5.8.



Gambar 0.8 Antarmuka Halaman Memilih Materi

Antarmuka halaman ketika pertama kali setelah masuk ke dalam sistem. Terdapat 3 tab halaman yang digambarkan menggunakan *Bottom Navigation Bar.* Setelah itu terdapat antarmuka halaman menjawab soal yang digambarkan pada Gambar 5.9.



Gambar 0.9 Antarmuka Halaman Menjawab Soal

Antarmuka halaman saat menjawab soal menggunakan suara. Dengan menekan tombol bulat di bagian bawah, maka kita diminta untuk menjawab soal menggunakan suara. Selanjutnya perancangan antarmuka halaman profil digambarkan pada Gmbar 5.10.

Gambar 0.10 Antarmuka Halaman Profil

Antarmuka halaman profil terdapat foto profil, nama profil, alamat email, dan materi apa saja yang sudah diikuti oleh pengguna. Selanjutnya terdapat halaman memilih kuis dgambarkan pada Gambar 5.11.

Gambar 0.11 Antarmuka Halaman Memilih Kuis

Antarmuka halaman memilih kuis terdapat judul kuis, deskripsi dan tombol-tombol untuk memulai kuis. Selanjutnya halaman splash screen terdapat pada Gambar 5.12.

Gambar 0.12 Antarmuka Halaman *Splashscreen*

Antarmuka halaman *splashscreen* akan selalu dimuat ketika aplikasi pertama kali dibuka. Terdapat logo dan nama aplikasi. Antarmuka halaman *welcome* akan digambarkan pada Gambar 5.13.

Gambar 0.13 Antarmuka Halaman *Welcome*

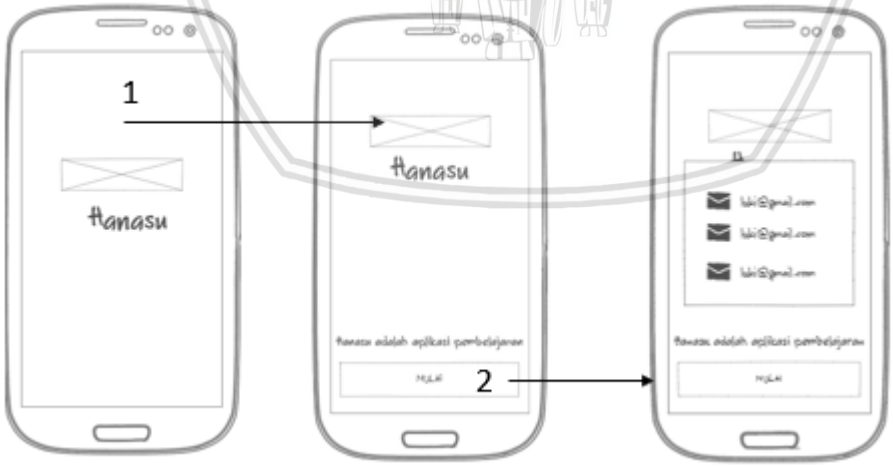
Antarmuka yang dimuat saat pertama kali setelah halaman *Splashscreen* dan belum terautentifikasi masuk ke dalam sistem. Antarmuka memilih akun google setelah menekan tombol mulai digambarkan pada Gambar 5.14.

Gambar 0.14 Antarmuka Memilih Akun Google

Antarmuka pada Gambar 5.14 merupakan antarmuka untuk memilih akun google mana yang akan digunakan selama menggunakan aplikasi.

Screenflow atau diagram alir screen setiap aktifitas digunakan untuk memudahkan implementasi agar mengerti setiap perpindahan halaman. *Screenflow* dari sistem penelitian yang dibuat dapat dilihat pada gambar dibawah ini :

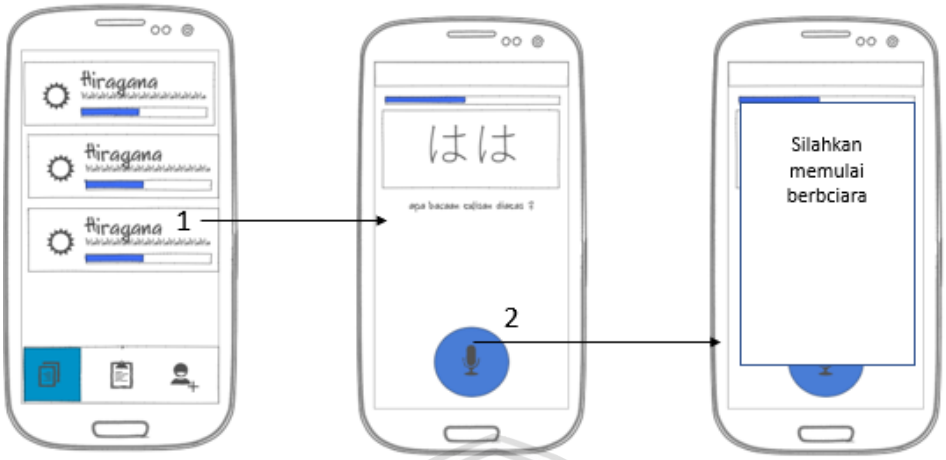
Screenflow pertama adalah *screenflow* yang dibuat untuk masuk ke dalam sistem menggunakan akun google, dapat dilihat pada Gambar 5.15



Gambar 0.15 *ScreenFlow* Masuk ke Dalam Sistem

Perpindahan antara halaman *splashscreen* menuju *welcome screen* terjadi otomatis bergantung pada kecepatan sistem membaca apakah perangkat yang digunakan sudah pernah masuk ke dalam sistem atau belum. Perpindahan kedua dari *welcome screen* hingga memilih akun google terjadi ketika kita menekan tombol pada halaman *welcome screen.*

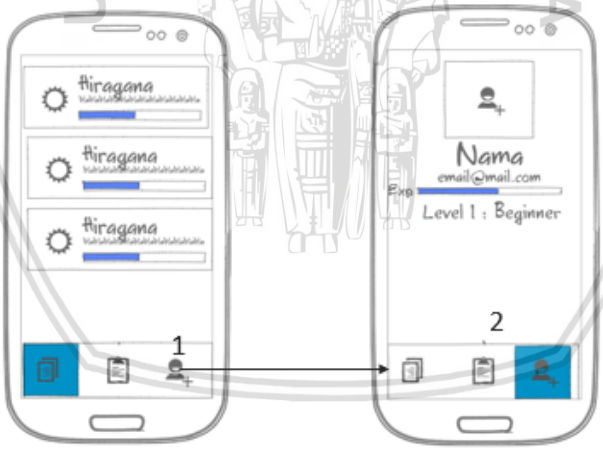
Screenflow kedua menunjukkan bagaimana proses untuk belajar bahasa jepang. Ditunjukkan pada Gambar 5.16.



Gambar 0.16 *Screenflow* Belajar Materi

Pada halaman memilih materi kita menekan salah satu materi yang membawa kita ke halaman belajar menjawab soal. Kemudian terdapat tulisan yang harus kita baca, setelah itu kita menekan tombol yang berada di bawah halaman, untuk muncul tampilan google speech "silahkan memulai bicara".

Screenflow ketiga yakni menunjukkan bagaimana cara melihat profil ditunjukkan pada Gambar 5.17.



Gambar 0.17 Melihat Profil

Pada halaman utama terdapat *BottomNavigationView* , di dalam *BottomNavigationView* terdapat 3 pilihan belajar, kuis, dan profil. Saat menekan pilihan nomor 1 maka akan dibawa ke halaman profil.

5.5.2 Sprint Kedua

Pada *sprint* kedua penulis membuat 1 antarmuka baru untuk soal yang memiliki gambar digambarkan pada Gambar 5.18.

Gambar 0.18 Antarmuka Menjawab Soal Bergambar

Antarmuka pada Gambar 5.18 hampir sama dengan antarmuka menjawab soal yang berbentuk tulisan, hanya saja soal tulisan berubah menjadi gambar. Untuk *screenflow* antarmuka menjawab soal bergambar sama seperti *screenflow* menjawab soal seperti biasa hanya saja tulisan berganti menjadi gambar.

5.6 Spesifikasi Perangkat Keras

Implementasi Hanasu ini dikembangkan menggunakan aplikasi Android Studio 3.1 di laptop ASUS A455L dan diuji coba di perangkat Android dengan spesifikasi masing-masing perangkat berada di dalam Tabel 5.5 dan Tabel 5.6

Tabel 0.5 Spesifikasi Perangkat Laptop ASUS A455L

Komponen	Spesifikasi
Model	ASUS A455L
Prosesor	Intel Core i3-4030U 1,9 GHz (A455LD)
RAM	DDR3 4 GB (A455LN)
GPU	NVDIA GeForce 930M

Tabel 0.6 Spesifikasi Perangkat Android

Komponen	Spesifikasi
Model	Samsung Galaxy J7 2017
Prosesor	Octa-core 1.6 GHz Cortex-A53
RAM	3 GB

Kapasitas Penyimpanan	16 GB

5.7 Spesifikasi Perangkat Lunak

Implementasi aplikasi Hanasu pengembangannya menggunakan perangkat berbasis windows, sedangkan pada perangkat bergerak berbasis Android. Spesifikasi perangkat lunak pada masing-masing perangkat ditunjukkan dalam Tabel 5.7 dan Tabel 5.8.

Tabel 0.7 Spesifikasi Perangkat Lunak Laptop ASUS A455L

Komponen	Spesifikasi
Sistem Operasi	Windows 10
IDE	Android Studio 3.1
Bahasa Pemrograman	Java 8

Tabel 0.8 Spesifikasi Perangkat Lunak Android

Komponen	Spesifikasi
Sistem Operasi	Android 6.0.1

5.8 Batasan Implementasi

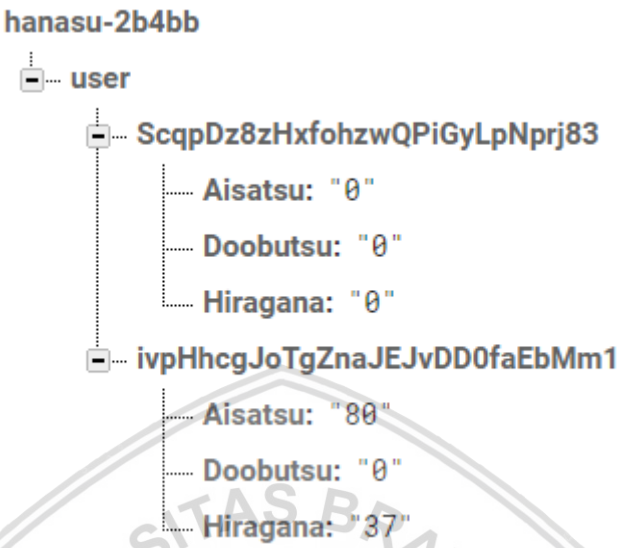
Dalam penelitian yang dilakukan oleh penulis, terdapat berapa batasan terkait dengan implementasi sistem yang akan dikembangkan. Batasan tersebut antara lain :

1. Aplikasi dikembangkan menggunakan *Smartphone* Android dengan versi OS minimal API 19.
2. Aplikasi dikembangkan menggunakan Bahasa Java.
3. Aplikasi dikembangkan menggunakan *Integrated Development Environtment* Android Studio 3.1
4. Aplikasi dikembangkan dengan menggunakan basis data *Firebase* dengan *library* versi 11.8.0.
5. Aplikasi dikembangkan dengan menggunakan *library google-play-service* 11.8.0
6. Aplikasi dikembangkan dengan menggunakan *picasso* versi 2.71828 untuk menampilkan gambar berdasarkan *url*.

5.9 Implementasi Basis Data

Implementasi basis data diimplementasikan menggunakan *firebase.* Menggunakan *Firebase* karena memiliki fitur *Real Time Database, Firebase Cloud Messaging* dan *Authentification* yang berintegasi dengan akun google. *Firebase* menggunakan konsep no-SQL

sehingga implementasi dari basis data yang digunakan menggunakan *JSON*. Implementasinya dapat dilihat pada gambar dibawah ini :



Gambar 0.19 JSON Basis Data Aplikasi Hanasu

Data yang disimpan ke dalam basis data *firebase* hanya data yang bersifat permanen, data yang diharapkan tidak akan hilang ketika pengguna berpindah perangkat.

5.10 Implementasi Kode Program

Implementasi kode program terkait dengan kebutuhan fungsionalitas sistem akan dilakukan dibagi menjadi 2 komponen besar, membaca tulisan Jepang dan mendengarkan tulisan Jepang. Karena penulis menggunakan Android Studio dalam pengimplementasian kode program, maka pembuatannya menggunakan adapter untuk menyatukan antara *Activity* dan *Fragment* sehingga *class-class* dapat digunakan kembali dengan *Activity* atau *Fragment* yang lain.

5.10.1 Implemenasi Kode Program Membaca Tulisan Jepang (Text-to-Speech)

Implementasi untuk membaca tulisan yang ada pada perangkat menggunakan *library* dari google bernama *TextToSpeech*, yang nantinya akan digunakan berulang-ulang untuk membaca banyak tulisan. Kode Program Membaca Tulisan Jepang akan dituliskan didalam tabel kode program 5.1 dibawah ini :

Kode Program 0.1 Membaca Tulisan Jepang (*Text-to-Speech*)

CharacterActivity.java

```
1  @Override
2  protected void onCreate(Bundle savedInstanceState) {
3      super.onCreate(savedInstanceState);
4      setContentView(R.layout.activity_character);
5
6      tts = new TextToSpeech(this, this);
7      speakOut();
8  }
9
```

```
10  @Override
11  public void onInit(int status) {
12      if (status == TextToSpeech.SUCCESS) {
13          int result = tts.setLanguage(Locale.JAPANESE);
14          if (result == TextToSpeech.LANG_MISSING_DATA
15                  || result ==
16  TextToSpeech.LANG_NOT_SUPPORTED) {
17              Log.e("TTS", "This Language is not supported");
18          } else {
19              speakOut();
20          }
21      } else {
22          Log.e("TTS", "Initilization Failed!");
23      }
24  }
25
26  @Override
27  public void onDestroy() {
28      if (tts != null) {
29          tts.stop();
30          tts.shutdown();
31      }
32      super.onDestroy();
33  }
34
35  private void speakOut() {
36      tts.speak(alphabet, TextToSpeech.QUEUE_FLUSH, null);
37  }
```

Pembahasan dari Kode Program 5.1 adalah sebagai berikut :

1. Baris 1 – 8 : Menginisialisasi object tts bertipe data *object TextToSpeech* yang berada di dalam *method onCreate*. *Method OnCreate* adalah *method* yang pertama kali dijalankan oleh *compiler*. Dan memanggil *method speakOut()* untuk mulai membaca tulisan yang ingin dimunculkan suaranya.

2. Baris 10 – 24 : Merupakan *method* yang harus di-*override* ketika menggunakan *library Text-to-Speech*, dengan mengecek apakah perangkat yang digunakan mendukung menggunakan *library Text-to-Speech* atau tidak, dan apakah perangkat yang digunakan mendukung menggunakan bahasa Jepang atau tidak.

3. Baris 25 – 33 : Merupakan *method* yang harus di-*override* ketika menggunakan *library Text-to-Speech*, dengan mematikan seluruh *service* yang digunakan oleh *library Text-to-Speech* agar tidak memakan daya perangkat yang lebih banyak ketikda tidak digunakan.

4. Baris 35 – 37 : Merupakan *method* yang digunakan untuk mengeluarkan suara berdasarkan tulisan yang akan dipilih, dengan memanggil *method speak().*

5.10.2 Implementasi Kode Program Berbicara Bahasa Jepang (Speech-to-Text)

Implementasi untuk berbicara bahasa Jepang pada perangkat menggunakan *library* dari *google* bernama *SpeechToText*, yang nantinya akan digunakan berulang-ulang untuk menjawab soal yang akan diberikan.

Kode Program 0.2 Kode Program Berbicara Bahasa Jepang (*Speech-to-Text*)

CourseActivity.java

```
1  public void SpeechInput() {
2      Intent intent = new
3  Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
4      intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL,
5              RecognizerIntent.LANGUAGE_MODEL_FREE_FORM);
6      intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE, "ja-
7  JP");
8      intent.putExtra(RecognizerIntent.EXTRA_PROMPT,
9              "Silahkan memulai bicara!");
10     try {
11         startActivityForResult(intent,
12 REQ_CODE_SPEECH_INPUT);
13     } catch (ActivityNotFoundException a) {
14         Toast.makeText(getApplicationContext(),
15                 "Maaf perangkat anda tidak dapat melakukan
16 speech",
17                 Toast.LENGTH_SHORT).show();
18     }
19 }
20
21 @Override
22 protected void onActivityResult(int requestCode, int
23 resultCode, Intent data) {
24     super.onActivityResult(requestCode, resultCode, data);
25
26     switch (requestCode) {
27         case REQ_CODE_SPEECH_INPUT: {
28             if (resultCode == RESULT_OK && null != data) {
29
30                 ArrayList<String> result = data
31
32 .getStringArrayListExtra(RecognizerIntent.EXTRA_RESULTS);
33                 suara = result.get(0);
34                 Toast.makeText(this, suara,
35 Toast.LENGTH_SHORT).show();
36             }
37             break;
38         }
39
40     }
41 }
42
```

Pembahasan dari Kode Program 5.2 adalah sebagai berikut :

1. Baris 1 – 19 : Membuat *Intent* baru dengan action *Action Recognize Speech,* dan mengatur bahwa bahasa yang akan digunakan adalah Bahasa Jepang dengan kode “ja-JP”. Setelah itu menggunakan *exception* untuk mengecek apakah perangkat yang digunakan mendukung akan adanya *Speech-to-Text* atau tidak.

2. Baris 21 – 41 : Merupakan *method* yang di-*override* ketika menggunakan *library Speech-To-Text*, untuk mendapatkan hasil setelah menggunakan *Intent* berupa *Speech-to-Text.* Apabila hasil dari berbicara sukses, maka kata-kata yang

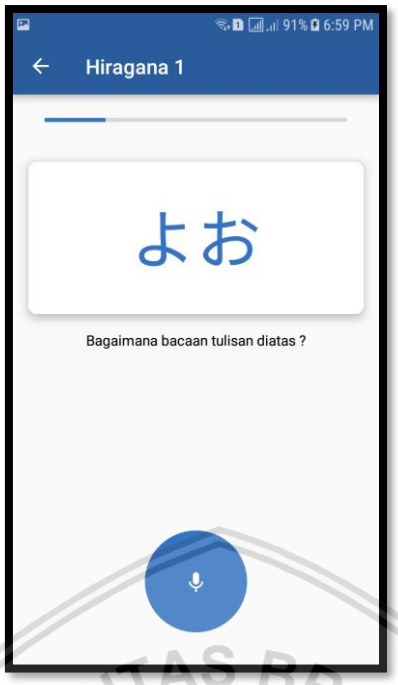
diucapkan dimasukkan ke dalam *variable* bertipe data *ArrayList* yang nantinya diambli dengan indeks ke-0.

5.11 Implementasi Antarmuka Pengguna

Implementasi antarmuka pengguna menampilkan antarmuka dari aplikasi Hanasu. Aplikasi Hanasu ini memiliki beberapa antarmuka pengguna diantaranya antarmuka halaman memilih materi (Gambar 5.15), antarmuka halaman menjawab soal di kuis (Gambar 5.16) , antarmuka halaman profil (Gambar 5.17), dan antarmuka halaman memilih kuis (Gambar 5.18). Lebih detailnya masing-masing halaman dijelaskan dibawah ini :

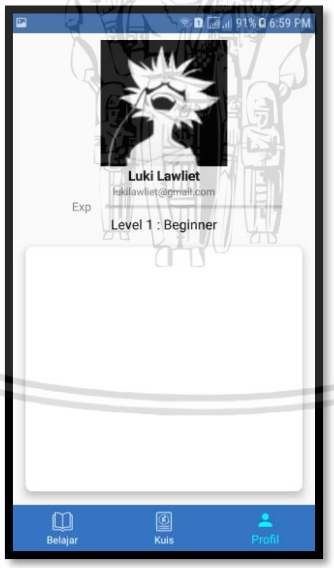
Gambar 0.20 Halaman Memilih Materi

Halaman ini menunjukkan seluruh materi yang ada pada aplikasi ini. Mulai dari yang paling mudah hingga menuju yang paling susah. Terdapat kata dalam bentuk alfabet dan Jepang sebagai judul materi. Terdapat deskripsi tentang materi, dan *Progress Bar* yang menunjukkan nilai dari 3 latihan yang sudah dibuat di dalam kuis.



Gambar 0.21 Halaman Menjawab Soal di Kuis

Halaman ini terdapat kata dalam huruf Jepang dan kita menjawab dengan menekan tombol bulat yang berada di bawah berdasarkan soal yang berada di bawah kotak. *Progress Bar* yang ada di atas menunjukkan perkembangan kuis yang sudah kita jawab.



Gambar 0.22 Halaman Profil

Pada halaman ini menunjukkan foto profile , nama akun, dan email yang sudah terintegrasi dengan akun *Google*. Pada halaman profil juga menampilkan *exp* saat ini yang menunjukkan perkembangan dari pembelajaran sang pengguna aplikasi. Terdapat macam-macam level yang akan berubah setiap kenaikan levelnya.

Gambar 0.23 Halaman Memilih Kuis

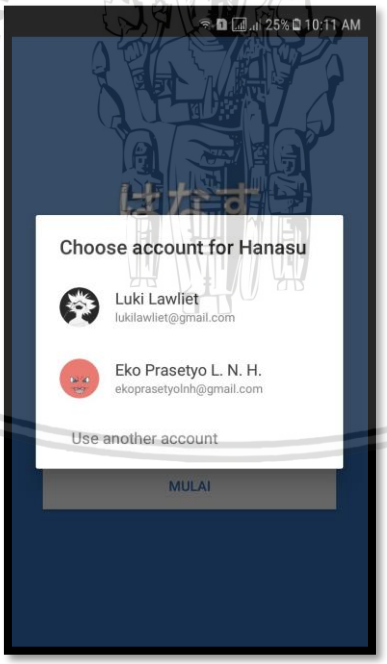
Halaman ini menunjukkan seluruh kuis yang dapat didapat, terdapat Judul kuis yang akan diambil, deskripsi kuis, dan tombol untuk memasuki halaman menjawab soal pada kuis.

Gambar 0.24 Halaman *Splash Screen*

Halaman ini adalah halaman yang akan selalu dimuat ketika aplikasi pertama kali dibuka. Terdapat logo dan nama aplikasi.

Gambar 0.25 Halaman *Welcome Screen*

Halaman ini akan dimuat saat pertama kali setelah *splash screen* dan belum terauthentifikasi masuk ke dalam sistem. Halaman ini akan mengarah ke saat dimana pengguna disuruh untuk memilih akun google.



Gambar 0.26 Halaman Memilih Akun Google

Halaman ini muncul disaat kita mengambil akun google, setelah menekan salah satu akun google di atas maka foto profil, nama, dan email akan menggunakan akun yang dipilih

BAB 6 PENGUJIAN

Pada bab ini berfokus pada pengujian aplikasi yang sudah dibuat apakah sudah memenuhi segala kebutuhan fungsionalitas dan apakah sudah berjalan dengan baik dan benar. Pengujian dibagi menjadi dua, pengujian fungsionalitas dan pengujian usabilitas.

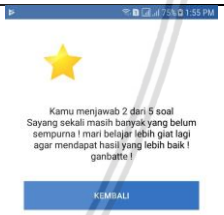
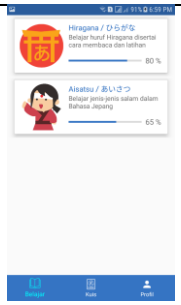
6.1 Pengujian Fungsionalitas

Pengujian fungsionalitas dilakukan menggunakan metode *blackbox testing*. Pengujian dilakukan dengan pertama-tama menentukan *test case* setelah itu *step by step*, kemudian kita berikan beberapa opsi yang mungkin terjadi beserta bukti tampilan dari program. Pengujian dilakukan setiap kali setelah *sprint* berakhir, oleh karena itu penelitian melakukan dua kali pengujian fungsionalitas.

Sprint pertama menghasilkan pengujian ditunjukkan pada Tabel 6.1.

Tabel 0.1 Pengujian *Black Box Sprint* Pertama

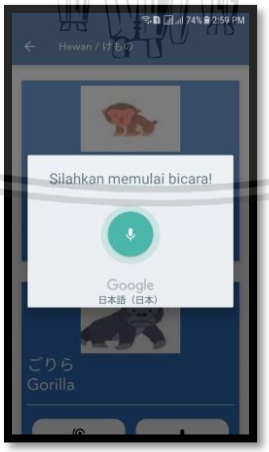
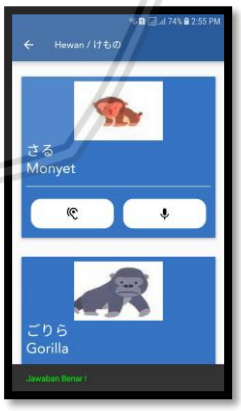
No	Test Case	Target	Test Execution	Expected Result	Validation
1	Menjawab soal berbentuk tulisan menggunakan lisan	Menjawab soal dengan benar	 Menekan tombol dengan icon *mic*	 Terdapat *alert* berwarna hijau dibawah halaman	Valid
		Menjawab soal dengan salah	 Menekan tombol dengan icon *mic*.	 Terdapat *alert* berwarna merah dibawah halaman	Valid

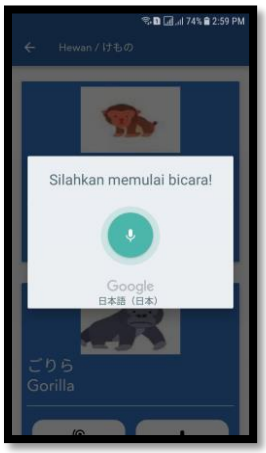
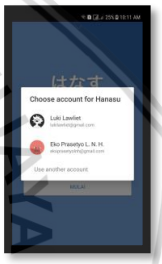
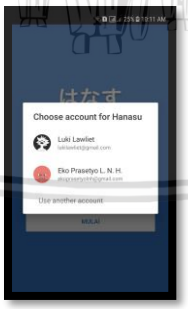
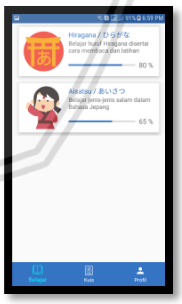
2	Mengikuti kuis	Menjawab soal dengan nilai 76 keatas dan dibawah 100	 Mengikuti kuis hingga selesai	 Muncul bintang 3 dengan kalimat selamat	Valid
		Menjawab soal dengan nilai 50 keatas dan diabawah 75	 Mengikuti kuis hingga selesai	 Muncul bintang 2 dengan kalimat saran	Valid
		Menjawab soal dengan nilai dibawah 49	 Mengikuti kuis hingga selesai	 Muncul bintang 1 dan kalimat penyemangat	Valid
3	Memberikan Bantuan	Bantuan muncul ketika aplikasi pertama kali diinstall			Valid

			Muncul bantuan berwarna merah	Bantuan hilang ketika ditutup	
4	Mendengarkan suara berdasarkan tulisan	Mendengarkan cara pelafalan kalimat dengan suara orang Jepang	 Menekan tombol dengan bentuk telinga	 Tidak ada perubahan tampilan namun ada suara muncul berdasarkan tulisan	Valid

Sprint kedua menguji masuk menggunakan akun Google dan menjawab soal berupa gambar. Apakah sudah bekerja dengan sesuai kebutuhan atau belum. Uji fungsionalitas pada *sprint* kedua dapat dilihat pada Tabel 6.2.

Tabel 0.2 Tabel Pengujian *Black Box Sprint* Kedua

No	Test Case	Target	Test Execution	Expected Result	Validation
1	Menjawab soal dengan gambar	Pengguna dapat menjawab soal bergambar	 Menekan tombol dengan gambar mic	 Jawaban benar dan muncul alert dengan tulisan berwarna hijau	Valid

			 Memilih salah satu akun google	 Jawaban salah dan muncul alert dengan tulisan berwarna merah	Valid
2	Masuk ke dalam sistem	Pengguna dapat masuk ke dalam sistem menggunakan akun Google	 Menekan tombol masuk	 Masuk ke halaman memilih akun google	Valid
			 Memilih salah satu akun google	 Masuk ke dalam sistem utama	Valid

6.2 Pengujian Usabilitas

Pengujian usabilitas dilakukan untuk menguji kebutuhan non-fungsionalitas sistem yang telah dikembangkan kepada beberapa responden secara langsung untuk mengetahui apakah sistem yang sudah kita bangun dapat digunakan dengan mudah atau tidak. Terdapat banyak metode untuk menguji usabilitas dari system. Salah satunya yang akan digunakan dalam penelitian ini yaitu *A/B Testing.*

6.2.1 A/B Testing

A/B Testing adalah pengujian yang membandingkan 2 atau lebih sistem untuk dicari mana yang paling optimal. *A/B Testing* mengijinkan peneliti membuat *traffic* dalam periode waktu tertentu. Sehingga perubahan kecil maupun besar dapat dilacak. Di dalam penelitian ini *A/B Testing* digunakan untuk menguji keefisienan saat belajar materi.

Penelitian ini menggunakan 5 responden sebagai stakeholder aplikasi yang akan menilai 2 tampilan yang berbeda saat menjawab soal, yakni tampilan antarmuka menjawab soal menggunakan lisan pada Tabel 6.3.

Tabel 0.3 Antarmuka Menjawab Soal yang diuji

No	Antarmuka Pertama	Antarmuka Kedua
1		

6.3 Pengujian Metodologi Belajar

Pada tahap ini penelitian menguji untuk membuktikan bahwa belajar dengan menggunakan aplikasi Hanasu lebih efisien daripada belajar menggunakan buku. Metode yang digunakan adalah metode *pre-test* dan *post-test design study.* Hasilnya nanti akan menunjukkan metode mana yang lebih cepat untuk dipahami oleh pengguna.

Metode ini pernah diujikan terhadap mahasiswa semester 5 dari kedokteran gigi di *Shiraz University of Medical Science* untuk membandingkan 2 metode berbeda dari pembelajaran (virtual dan tradisional) dalam pembelajaran tentang kedokteran gigi. Dengan hasil desain virtual baru lebih layak dan efektif dibanding pembelajaran tradisional (Fariborz Moazami, 2014). Langkah-langkah yang diambil untuk menguji metode pembelajaran mana yang lebih efektif adalah sebagai berikut :

1. **Menseleksi Kriteria Pengguna**

   Kriteria pengguna yang dilakukan pada penelitin ini adalah seluruh siswa yang memiliki minat untuk belajar Bahasa Jepang dan minimal sudah memiliki dasar mengenali huruf Bahasa Jepang (*Hiragana*).

Didapatkan hasil dari beberapa responden yang memiliki minat untuk belajar Bahasa Jepang denga data responden yang ditampilkan pada Tabel 6.4.

Tabel 0.4 Data Responden

No	Nama	Umur
1	Maulidya Selene S. P.	17
2	Nanda Aulia	16
3	Esa Ardhianty	17
4	Rhesa Pratama	17
5	Dian Fiqi Amalia	17

2. **Koleksi Data**

Peneliti wajib mengkontrol variabel pengujian dalam pendekatannya, diantaranya :

- Materi yang diujikan hanya akan menggunakan huruf hiragana.
- Materi yang diujikan hanya sebatas tingkat kelas 2 SMA.
- Tidak ada peserta yang memiliki tingkatan lebih tinggi dari yang lain.
- Jumlah soal yang diujikan antara tradisional dan menggunakan aplikasi harus masih 1 bidang yang sama.

3. **Metode Pembelajaran**

Metode pembelajaran untuk menguji performa dari keduanya digambarkan dalam Tabel 6.5.

Tabel 0.5 *Task Scenario* Metode Belajar

No Urutan	Scenario
1	Peserta diberikan materi berupa kertas dan mempelajari isi dari materi tersebut dalam waktu 10 menit.
2	Peserta memasuki ruangan yang berbeda untuk diberikan pertanyaan sesuai materi yang sudah diajarkan.
3	Fasilitator menilai jumlah pertanyaan yang benar dan yang salah.
4	Kemudian diberikan aplikasi Hanasu yang berisikan materi dengan tema yang sama seperti sebelumnya, dan mereka mempelajari materi tersebut dalam waktu 10 menit.
5	Peserta memasuki ruangan yang berbeda untuk diberikan pertanyaan sesuai materi yang sudah diajarkan saat menggunakan aplikasi.
6	Proses diulang dengan materi pertama membahas nama-nama hewan dalam Bahasa Jepang dan kedua membahas

	tentang pernyataan yang paling sering digunakan dalam Bahasa Jepang

Pertanyaan yang akan diajukan akan ditunjukkan dalam Tabel 6.6.

Tabel 0.6 Soal untuk Diujikan Pada Iterasi Pertama

No	Materi di Kertas	Materi di Aplikasi
1	とり/ Tori /Burung	ねこ/ Neko / Kucing
2	いぬ/ Inu / Anjing	さかな / Sakana / Ikan
3	かめ / Kame / Kura-kura	うし / Ushi / Sapi
4	かえる / Kaeru / Katak	ぞう / Zou / Gajah
5	うま / Uma / Kuda	ねずみ / Nezumi / Tikus
6	ぶた / Buta / Babi	へび / Hebi / Ular
7	いるか / Iruka / Lumba-lumba	さめ / Same / Hiu
8	らいおん / Raion / Singa	おおかみ / Ookami / Serigala
9	きつね / Kitsune / Rubah	くま / Kuma / Beruang
10	いか / Ika / Cumi-cumi	みみず / Mimizu / Cacing
11	やもり / Yamori / Cicak	うさぎ / Usagi / Kelinci
12	かい / Kai / Kerang	ほたる / Hotaru / Kunang-kunang
13	やぎ / Yagi / Kambing	はえ / Hae / Lalat
14	くも / Kumo / Laba-laba	さる / Saru / Monyet
15	あり / Ari / Semut	えび / Ebi / Udang
16	くじら / Kujira / Paus	うなぎ / Unagi / Belut
17	おんどり / Ondori / Ayam Jantan	めんどり / Mendori / Ayam Betina
18	ひつじ / Hitsuji / Domba	からす / Karasu / Gagak
19	たか / Taka / Elang	きりん / Kirin / Jerapah
20	わに / Wani / Buaya	りす/ Risu / Tupai

Pada iterasi kedua pertanyaan dibuat lebih panjang dan lebih interaktif. Materi dapat dilihat pada Tabel 6.7.

Tabel 0.7 Soal untuk Diujikan Pada Peserta Iterasi Kedua

No	Materi di Kertas	Mater di Aplikasi
1	おはようございます / Ohayou Gozaimasu / Selamat Pagi	こんにちは / Konnichiwa / Selamat Siang
2	こんばんは / Konbanwa / Selamat Malam	おやすみなさい / Oyasumi nasai / Selamat Tidur
3	さようなら / Sayounara / Selamat Tinggal	またあした / Mata Ashita / Sampai Jumpa Besok
4	またらいしゅう / Mata Raishuu / Sampai Jumpa Minggu Depan	またらいげつ / Mata Raigetsu / Sampai Jumpa Bulan Depan
5	またらいねん / Mata Rainen / Sampai Jumpa Tahun Depan	ありがとうございます / Arigato Gozaimasu / Terima Kasih
6	すみません / Sumimasen / Permisi	どういたしまして / Douita Shimashte / Sama-sama
7	おねがいします / Onegaishimasu / Minta Tolong	おげんきですか / Ogenki Desuka ? / Apa Kabar ?
8	ごめんなさい/ Gomennasai / Minta maaf	いらっしゃいませ / Irasshaimase / Selamat Datang
9	きをつけてください / Kiwotsuketekudasai / Hati-hati ya	ちょっとまってください / Chottomatekudasai / Tolong tunggu sebentar
10	いただきます / Itadakimasu / Selamat Makan	ただいま / Tadaima / Diucapkan Ketika Pulang Ke Rumah

4. **Hasil**

Setelah diberikan materi dan pertanyaan hasil dari pengujian diatas, hasil disajikan dalam bentuk persentase jumlah pertanyaan yang benar dan yang salah. Hasil dari pengujian dapat dilihat pada Tabel 6.8.

Tabel 0.8 Hasil Pengujian Performa Sistem

No	Nama	Jumlah Benar Dengan Materi Buku		Jumlah Benar Dengan Materi Aplikasi Hanasu	
		Test 1	Test 2	Test 1	Test2
1	Maulidya Selene	13/20	10/10	16/20	10/10
2	Nanda Aulia	17/20	6/10	17/20	9/10
3	Esa Ardhianty	4/20	2/10	5/20	3/10

4	Rhesa Pratama	15/20	9/10	15/20	9/10
5	Dian Fiqi Amalia	12/20	6/10	10/20	6/10

6.4 Analisis Hasil Pengujian

Penelitian ini menggunakan 5 responden yang memiliki keterkaitan dengan aplikasi sebagai penguji untuk menguji sistem yang sudah dibuat. Parameter yang diuji diambil berdasarkan kebutuhan non-fungsionalitas yang sudah dibuat dari berupa kesesuaian dan kecepatan, dengan detail seperti dibawah ini :

- Kecepatan : merupakan waktu yang ditempuh untuk menyelesaikan 3 buah pertanyaan di dalam sistem.
- Kesesuaian : merupakan kenyamanan dalam penggunaan system.

Parameter di atas dipilih karena penelitian ini menginginkan aplikasi yang mudah dan cepat digunakan. Pengujian dilakukan menggunakan perangkat yang sama, dan dengan data soal yang sama. Sehingga dihasilkan dari *A/B Testing* dari aktifitas belajar materi dan menjawab soal menggunakan lisan dapat dilihat pada Tabel 6.9 dan ditarik lebih jauh menggunakan klasifikasi tingkat kepuasan pengguna pada Tabel 6.10.

Tabel 0.9 Hasil Dari Pengujian Terhadap Responden

No	Parameter	Antarmuka pertama	Antarmuka kedua
1	Kesesuaian	80 %	20 %
2	Kecepatan	9.600 detik	11,600 detik

Tabel 0.10 Klasifikasi Tingkat Kepuasan Pengguna

No	Interval Persentase Kepuasan Pengguna	Tingkat Kepuasan Pengguna
1	0 – 20%	Sangat tidak memuaskan
2	21 – 40%	Tidak memuaskan
3	41 – 60%	Netral
4	61 – 80%	Memuaskan
5	81 – 100%	Sangat memuaskan

Hasil pada pengujian usabilitas pada Tabel 6.10 disertai pula dengan alasan yang dilampirkan pada lampiran B. Sehingga didapatkan antarmuka yang digunakan untuk menjawab soal menggunakan lisan dipilih adalah antarmuka pertama, dan antarmuka pertama memiliki tingkat kepuasan pengguna memuaskan.

Rata-rata dari hasil dari pengujian performa sistem dapat dilihat pada Tabel 6.11.

Tabel 0.11 Rata-rata Hasil Pengujian Performa

Persentase Jumlah Benar Dengan Materi Buku		Persentase Jumlah Benar Dengan Materi Aplikasi Hanasu	
Test 1	**Test 2**	**Test 1**	**Test2**
61%	66%	63%	74%

Berdasarkan waktu 10 menit yang diujikan. Hasil dari rata-rata pada Tabel 6.11 menunjukkan bahwa pada test pertama (menghafal nama-nama hewan dalam Bahasa Jepang) meningkat 2% , dan test kedua (menghafal kosakata yang sering diucapkan dalam Bahasa Jepang) meningkat 8%. Dapat disimpulkan bahwa belajar menggunakan aplikasi Hanasu dapat membuat pembelajaran lebih cepat dalam berbicara Bahasa Jepang

BAB 7 PENUTUP

Setelah melakukan penelitian hingga akhir, diambil kesimpulan dan saran untuk sistem yang lebih baik lagi ke depannya.

7.1 Kesimpulan

Berdasarkan hasil analisis kebutuhan, perancangan, implementasi dan pengujian yang dilakukan sebelumnya pada penelitian ini, maka dapat diambil kesimpulan seperti dibawah ini :

1. Analisis kebutuhan dalam aplikasi Hanasu diawali dengan mendeskripsikan sistem yang akan dibuat dan lingkungan sistem. Hasil dari analisis kebutuhan aplikasi Hanasu berupa 8 *backlog* produk yaitu Sistem harus mampu menyediakan tempat untuk dapat masuk ke dalam sistem, Sistem harus mampu menyediakan mekanisme untuk menjawab soal menggunakan suara, Sistem harus mampu menyediakan mekanisme penilaian benar atau salah, Sistem harus mampu menyediakan fungsi bantuan untuk mempermudah pengguna, Sistem harus mampu menyediakan mekanisme mengeluarkan suara apablia menekan salah satu materi, Sistem harus menyediakan materi tentang huruf Bahasa Jepang, Sistem harus mampu menyediakan materi tentang kosakata Bahasa Jepang, dan Sistem harus mampu menyediakan soal berupa gambar.
2. Perancangan aplikasi Hanasu diawali dengan menggunakan Diagram *class* dengan *class* yang sudah siap diimplementasi. Dengan beberapa jumlah jenis *class* seperti *class activity, class fragment, class controller, class adapter* dan *class model.* Perancangan aplikasi Hanasu juga menghasilkan perancangan basis data dimana terdapat entitas *character, lesson* dan *profile.* Perancangan aplikasi Hanasu juga menghasilkan beberapa antarmuka diantaranya antarmuka memilih materi, antarmuka menjawab soal, antarmuka profil, antarmuka memilih kuis, antarmuka *splashscreen,* antarmuka selamat datang, dan antarmuka memilih akun *google*.
3. Implementasi aplikasi Hanasu menggunakan Android Studio dengan menggunakan bahasa Java. Hasilnya adalah beberapa *class* dengan format .java sebagai logika pemrograman dan .xml sebagai tampilan yang akan diliaht oleh pengguna secara langsung. Basis data yang dihasilkan berupa *tree* dengan 1 root bernama *user,* dengan branch nama–nama materi yang diujikan di dalam aplikasi dengan *value* angka 0 – 100.
4. Pengujian yang dilakukan pada penelitian ini terdapat 3 pengujian, pertama menguji fungsionalitas untuk menjaga aplikasi Hanasu tetap bekerja sesuai fungsionalitas. Hasilnya seluruh fitur yang berada pada *backlog* produk dapat terimplementasi tanpa terkecuali. Pengujian kedua adalah menguji usabilitas sistem Hanasu menggunakan *A/B Testing* dimana hasilnya kesesuaian antarmuka pertama lebih banyak digemari dengan unggul 60% dari antarmuka kedua dan dari segi kecepatan antarmuka pertama lebih cepat 2 detik dibandingkan antarmuka kedua. Setelah itu pengujian performa menggunakan *post-test* yang hasilnya pada test pertama aplikasi hanasu unggul 2% dari metode belajar menggunakan buku dan aplikasi hanasu unggul 8% dari metode belajar

meggunakan buku. Dapat disimpulkan bahwa belajar dengan menggunakan aplikasi Hanasu dapat mempercepat proses pembelajaran Bahasa Jepang.

7.2 Saran

Saran yang dapat diberikan untuk penelitian selanjutnya agar menghasilkan sistem yang lebih baik lagi adalah sebagai berikut :

1. Sistem dapat digunakan saat keadaan offline.
2. Sistem dapat menggunakan library *voice recognition* yang lain yang dapat membedakan huruf hiragana / katakana / kanji dalam sekali waktu.

Sistem dapat membuat tampilan level lebih variatif dan mendapatkan reward yang lebih menarik.

DAFTAR PUSTAKA

Moazami, Fariborz., Ehsan Bahrampour., Mohammad Reza A., Farzad Jahedi., dan Marzieh Moattari. 2014. "*Comparing two methods of education (virtual versus traditional) on learning of Iranian denal students: a post-test only design study*". BMC Medical Education BioMed Central dalam https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3975717/

Sommerville, Ian. 2004. "*Software Engineering , 7th Edition. Chapter 6*". International computer science series, Addison-Wesley 2007, ISBN 9780321313799, pp. I-XXIII, 1-840

Nielsen, Jakob. 2010. "*Interviewing User*". Diakses pada 10 April 2018. https://www.nngroup.com/articles/interviewing-users/

Budiu, Raluce. 2017. "*Quantitative vs Qualitative Usability Testing*". Diakses pada 10 April 2018. https://www.nngroup.com/articles/quant-vs-qual/

Levison, Mark. 2013. "*Choosing A Sprint Length – Shorter Trumps Longer*" . Diakses pada 10 April 2018. https://agilepainrelief.com/notesfromatooluser/2013/07/choosing-sprint-length-shorter-trumps-longer.html#.WtOgNohubIU

Verwijs, Christiaan. 2017. "*8 Useful Strategies for Splitting Large User Stories and a Cheatsheet*". Diakses pada 10 April 2018. https://blog.agilistic.nl/8-useful-strategies-for-splitting-large-user-stories-and-a-cheatsheet/

Wake, Bill. 2003. "*INVEST in Good Stories, and SMART Tasks*". Diakses pada 10 April 2018. https://xp123.com/articles/invest-in-good-stories-and-smart-tasks/

Jakarta, Japan Foundation. 2012. "*Survei Lembaga Pendidikan Bahasa Jepang di Indonesia tahun 2012*". Diakses pada 1 Oktober 2017. http://mayantara.sch.id/artikel/survei-lembaga-pendidikan-bahasa-jepang-di-indonesia-tahun-2012.htm

Tadashi, Ogawa. 2015. "*Pembelajar Bahasa Jepang Terbanyak: Indonesia Peringkat Kedua Dunia*". Diakses pada 1 Oktober 2017. http://kabar24.bisnis.com/read/20151026/255/486045/pembelajar-bahasa-jepang-terbanyak-indonesia-peringkat-kedua-dunia

Jakarta, Japan Foundation. 2013. "*Nuansa*". Diakses pada 1 Oktober 2017. http://mayantara.sch.id/artikel/survei-lembaga-pendidikan-bahasa-jepang-di-indonesia-tahun-2012.htm

Izza, Jamalul. 2016. "*2016, Pengguna Internet di Indonesia Capai 132 Juta".* Diakses pada 1 Oktober 2017. https://tekno.kompas.com/read/2016/10/24/15064727/2016.pengguna.internet.di.indonesia.capai.132.juta.

Arno, Abdul Kadir. 2014. "*Metode Pembelajaran Praktik*". Diakses pada 2 Oktober 2017. https://abdulkadirarno.wordpress.com/2014/06/05/metode-pembelajaran-praktik/

Pica, Teresa. 2005. "*Classroom Learning, Teaching, and Research: A Task-Based Perspective*". *The Modern Language Journal, 89, iii, (2005) 0026-7902/05/339-352*

Advess, akhsanov-HRD. 2016. "*Apa itu Scrum? Apa Saja Manfaatnya?".* Diakses pada 2 Desember 2017. https://www.kompasiana.com/advessbusiness/apa-itu-scrum-apa-saja-manfaatnya_5807261545afbd38183aab3a

Nielsen, Jakob. 2012. *"Usability 101: Introduction to Usability".* Diakses pada 3 Februari 2018. https://www.nngroup.com/articles/usability-101-introduction-to-usability/

Jacobson, Ivar. Spence, Ian. Bittner, Kurt. "*USE-CASE 2.0 The Guide to Succeeding with Use Cases*" *Ivar Jacob International*

Vincent, James. 2017. "*Google is using machine learning to sort good apps from bad on the Play Store*" Diakses pada 30 April 2018. https://www.theverge.com/2017/7/12/15958372/google-machine-learning-ai-app-store-malware-security